The Virgin Computer Games Series

Series Editor: Tim Hartnell

GAMES FOR YOUR ATARI 600 XL

£££££'s of entertaining games for only £2.95
Compatible 400/800

By Gary Ryan and Cliff McConnell

GAMES FOR YOUR ATARI 600 XL

By
Gary Ryan
and
Cliff McConnell

Virgin Books

First published in Great Britain in 1984 by Virgin Books Ltd,
61-63 Portobello Road, London W11 3DD.

Copyright © 1984 Interface/Virgin Books

ISBN 0 86369 037 8

All rights reserved. No part of this book may be reproduced in any form or by any means without prior permission from the publisher.

Printed and bound in Great Britain by Richard Clay
(The Chaucer Press) Ltd, Suffolk.

Production services by Book Production Consultants,
Cambridge.

Designed by Ray Hyden.

Illustrated by Sue Walliker.

Typeset by QV Typesetting.

Distributed by Arrow Books.

TIM HARTNELL — THE SERIES EDITOR

Tim Hartnell is the most widely-published computer author in the world. Founder of the National ZX Users' Club, and founding editor of *ZX Computing* magazine, Tim has been involved over the years in a wide variety of computer activities. His published works include *The Personal Computer Guide* (Virgin Books) and *The Giant Book of Computer Games* (Fontana).

GARY RYAN & CLIFF McCONNELL — THE AUTHORS

Gary Ryan and Cliff McConnell are 16-year-old schoolboys who live in Canberra, the capital of Australia. When they're not computing, Gary and Cliff enjoy bike-riding, playing cricket and swimming in the city's artificial lake, Lake Burley Griffin. Cliff breeds budgerigars as a hobby and both he and Gary plan to join the public service "to run the country' when they finish studies. Gary thinks that, if running the country doesn't come off, he'll join the Air Force.

SUE WALLIKER — THE ILLUSTRATOR

Sue Walliker is a freelance illustrator.

CONTENTS

Editor's Introduction 6

Author's Introduction 7

Special Programming Notes 8
Traffic Panic 9
Breakthrough 17
Shootout 25
Tower of Flames 29
Asteroid Storm 36
Blockout 43
Space Attack 47
Gobbler 55
Grand Prix 62
Escape 66
Time Typist 69
Galax Attack 71
Chopper Mission 83
Reversi 94
Mars Lander 99
How to Write Better Programs 104
Glossary 109
Bibliography 121

Editor's Introduction

Typing in a computer program is like opening an unknown door. You do not know until you actually open the door — or, in our case, run the program — what experience is waiting for you. Of course, the sign on the door has given you some indication, but nothing can equal first-hand experience.

You do not know precisely what experiences are waiting for you in the great programs in this book. Of course, if the introduction says you're entering a space game, it's very likely the program won't play 'Guess My Number' when you get it up and running. But the listing rarely hints at the computer's game-playing strategy, or the screen display, or the fun that is waiting for you.

This book has a number of unknown doors — doors leading into outer space and into the fiendish worlds of computer intelligence, wizards and Adventure.

We've provided the doors...and the keys. All you have to do to turn the lock is type in the program, and run it. Whatever you find behind each door, I guarantee you won't be disappointed.

Tim Hartnell
Series editor
London
March 1984

Authors' Introduction

After many months of planning, designing and hard programming, we bring you the book you have been waiting for. Everybody prefers a particular type of game, and we have tried to cater to all interests. There are board games, card games, strategy games and, best of all, fast-paced arcade games. And unlike other program listings, all of our games are efficiently programmed, fast, colourful, and fun to play.

So power up your system, type in these games, and let the fun begin!

Gary Ryan and Cliff McConnell
Canberra, Australia
March 1984

SPECIAL PROGRAMMING NOTES

While typing in these games, there are some special symbols you will need to know.

When you see the " | " symbol it means push the INVERSE key. When you see the "↑" symbol it means press the CONTROL key and hold it down. When you see the "←" key it means take your finger off the CONTROL key.

For a system using a DOS other than ATARI DOS (DOS 2), more memory than the stated amount may be needed. This is only if the DOS in use takes up more memory. ('Asteroid Storm', for example, will not run with a DOS that takes up more memory than Atari's DOS 2. Some other games in this book — those which use machine code — will not work instantly with non-Atari versions of DOS. If you discover that a program will not work with your DOS, load in the game, type POKE 9, 0 and press RETURN, then press SYSTEM RESET and run the game.)

As you will see, each game in this book makes use of some, or all, of the following Atari features: hi-res graphics, redefinable character sets, sound, player-missile graphics, display-list interrupts, vertical-blank interrupts, colour and mixed graphics modes. Most have machine language, and many make extensive use of it.

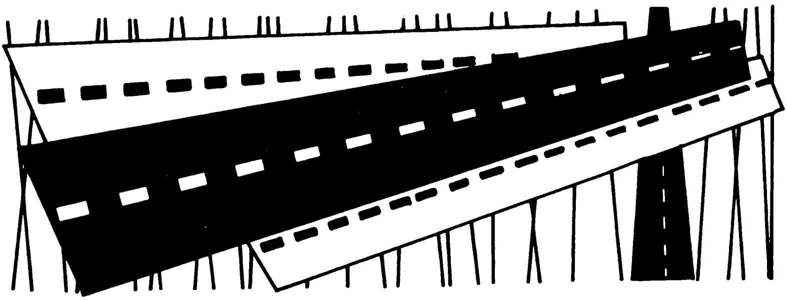
TRAFFIC PANIC

(16K CASSETTE, 24K DISK)

It is the year 2039 and, because of ridiculously overcrowded roads and the lack of available land, highways are being built one above the other. You are at the bottom of one of these 'super-highways' but you long to see the view from the top. For you to satisfy this ambition, you must climb up five levels using the abandoned ladders. However, each level is busy with traffic and the only way to avoid being flattened is by jumping over the oncoming cars.

After the program initialises the machine language, the title screen will be shown. Press SELECT to change the stage you start on. Press START to begin the game.

You are given three lives. After you have lost them all, the program returns to the title screen and displays the high score. Push a joystick in port 1 up to climb a ladder (if you are underneath one) and push it down to descend a ladder. Push it left to move left, and right to move right. Press the fire button to jump on the same spot, and press the fire button with the joystick to jump left or right.



```
10 REM **TRAFFIC PANIC
20 REM **Written by Cliff McConnell
30 REM **ALL REMS MAY BE OMITTED
60 DIM S(5),N(5),X1(5),Y1(5),X2(5),Y2(
5)
110 TOP=PEEK(740):PMBASE=(TOP-8)*256:S
TART=(TOP-12)*256:POKE 106,TOP:POKE 0,
PMBASE/256:GRAPHICS 0
115 POKE 106,TOP-3:GRAPHICS 0:POKE 106
,TOP-5:CHSET=(TOP-14)*256
120 VBLANK=START:DLI=START+604:USER=ST
ART+645
130 FOR K=0 TO 5:X1(K)=0:Y1(K)=0:X2(K)
=0:Y2(K)=0:NEXT K
140 GRAPHICS 17
150 POSITION 6,5:? #6;"TRAFFIC":POSITI
ON 7,7:? #6;"panic"
210 GOTO 510
300 REM PAGE 6 DATA
310 DATA 0,0,0,0,28,28,62,62,255,255,2
55,66,0,0,0,0,1,130,3,132,5,134,133,15
3,173,193
311 DATA 248,12,32,52,142,162,182,202,
239,3,23,43,151,171,191,211,230,250,14
,34,11,15,2,8,7,14
312 DATA 0,0,0,24,24,60,60,126,66,66,1
26,60,60,24,24,0,0,0,0,24,24,60,36,102
,102,102
313 DATA 102,36,60,24,24,0,0,0,0,24,24
,60,60,126,66,66,126,60,60,24,24,0,0,2
4,24,60
314 DATA 60,126,66,66,126,60,60,24,24,
0,0,0,100,-1
400 REM MACHINE LANGUAGE
410 DATA 169,4,141,128,6,162,3,189,22,
6,157,0,208,202,16,247,173,139,6,240,3
,76,95,228,198,207
411 DATA 208,53,165,208,133,207,160,18
,177,88,201,16,208,6,169,25,145,88,208
,27,56,233,1,145,88,201
```

```
412 DATA 16,208,26,136,177,88,201,16,2
08,19,169,1,141,139,6,141,140,6,76,95,
228,136,177,88,56,233
413 DATA 1,145,88,162,5,189,16,6,41,11
2,74,74,74,74,133,203,189,16,6,41,15,2
4,125,46,6,157
414 DATA 46,6,201,16,144,7,41,15,157,4
6,6,230,203,138,10,10,168,24,105,4,133
,204,189,16,6,48
415 DATA 16,185,22,6,24,101,203,153,22
,6,200,196,204,208,242,240,14,185,22,6
,56,229,203,153,22,6
416 DATA 200,196,204,208,242,202,16,17
5,173,131,6,208,127,173,137,6,208,31,1
73,16,208,208,84,169,1,141
417 DATA 138,6,173,0,211,106,106,106,1
76,5,162,0,142,138,6,106,176,5,162,2,1
42,138,6,238,137,6
418 DATA 173,137,6,201,21,176,5,206,13
3,6,208,12,238,133,6,201,40,208,5,169,
0,141,137,6,173,132
419 DATA 6,24,109,138,6,56,233,1,141,1
32,6,201,48,176,3,238,132,6,201,210,14
4,3,206,132,6,169
420 DATA 1,208,63,173,0,211,106,106,10
6,176,27,206,132,6,173,132,6,201,48,17
6,3,238,132,6,206,130
421 DATA 6,208,7,169,3,141,130,6,208,3
2,208,38,106,176,35,238,132,6,173,132,
6,201,210,144,3,206
422 DATA 132,6,206,130,6,208,17,169,3,
141,130,6,208,2,208,81,173,135,6,73,1,
141,135,6,173,0
423 DATA 211,106,176,83,173,131,6,208,
33,174,129,6,189,116,6,205,132,6,208,1
2,169,0,141,134,6,169
424 DATA 1,141,131,6,208,10,189,122,6,
205,132,6,208,47,240,234,206,133,6,238
,134,6,206,136,6,208
425 DATA 13,169,4,141,136,6,173,135,6,
73, 1, 141, 135, 6, 173, 134, 6, 201, 32, 208, 10
0, 169, 0, 141, 131, 6
426 DATA 141, 135, 6, 238, 129, 6, 24, 144, 86
, 106, 176, 83, 173, 131, 6, 208, 44, 174, 129, 6
, 240, 73, 202, 189, 116, 6
427 DATA 205, 132, 6, 208, 20, 142, 129, 6, 16
9, 32, 141, 134, 6, 169, 2, 141, 135, 6, 169, 1, 1
41, 131, 6, 208, 10, 189
428 DATA 122, 6, 205, 132, 6, 208, 36, 240, 22
6, 206, 136, 6, 208, 13, 169, 4, 141, 136, 6, 173
, 135, 6, 73, 1, 141, 135
429 DATA 6, 238, 133, 6, 206, 134, 6, 208, 8, 1
69, 0, 141, 131, 6, 141, 135, 6, 173, 135, 6, 10,
10, 10, 10, 170, 165
430 DATA 0, 24, 105, 3, 133, 204, 169, 0, 133,
203, 169, 15, 133, 205, 172, 133, 6, 189, 52, 6,
145, 203, 200, 232, 198, 205
431 DATA 16, 245, 173, 132, 6, 141, 4, 208, 56
, 233, 2, 141, 5, 208, 56, 233, 2, 141, 6, 208, 56
, 233, 2, 141, 7, 208
432 DATA 162, 0, 173, 133, 6, 205, 141, 6, 240
, 4, 141, 141, 6, 170, 142, 0, 210, 232, 142, 2, 2
10, 169, 168, 141, 1, 210
433 DATA 141, 3, 210, 76, 95, 228, 72, 138, 72
, 174, 128, 6, 189, 22, 6, 141, 0, 208, 189, 23, 6
, 141, 1, 208, 189, 24
434 DATA 6, 141, 2, 208, 189, 25, 6, 141, 3, 20
8, 232, 232, 232, 232, 142, 128, 6, 104, 170, 10
4, 64, 104, 169, 0, 141, 139
435 DATA 6, 173, 133, 6, 201, 34, 176, 6, 169,
1, 141, 139, 6, 96, 173, 9, 208, 24, 109, 10, 208
, 24, 109, 140, 6, 240
436 DATA 230, 141, 139, 6- 96, 0, -1
440 REM CHARACTER SET DATA
442 DATA 255, 165, 255, 0, 0, 0, 0, 0
444 DATA 99, 127, 127, 99, 99, 127, 127, 99
446 DATA 255, 165, 255, 99, 99, 127, 127, 99
448 DATA 0, 12, 30, 63, 63, 30, 12, 0
450 REM DATA FOR CARS
460 DATA 28, 28, 62, 62, 255, 255, 255, 66
```

```
470 DATA 56,56,124,124,255,255,255,66
510 TRAP 1020
520 REM READ IN PAGE 6 DATA
1010 FOR T=1536 TO 9999:READ A:POKE T,
A:NEXT T
1020 TRAP 1520
1030 REM READ IN ML
1510 FOR T=START TO START+9999:READ A:
POKE T,A:NEXT T
1520 POKE 54279,PMBASE/256:POKE 53277,
3:POKE 559,62:POP
1524 FOR T=0 TO 511:POKE CHSET+T,PEEK(
57344+T):NEXT T
1527 FOR T=0 TO 31:READ A:POKE CHSET+2
4+T,A:NEXT T
1530 FOR K=40 TO 227 STEP 64:RESTORE 4
60:FOR T=0 TO 7:READ A:POKE PMBASE+102
4+K+T,A:POKE PMBASE+1280+K+T,A
1540 POKE PMBASE+1536+K+T,A:POKE PMBAS
E+1792+K+T,A:NEXT T:NEXT K
1550 FOR K=72 TO 227 STEP 64:RESTORE 4
70:FOR T=0 TO 7:READ A:POKE PMBASE+102
4+K+T,A:POKE PMBASE+1280+K+T,A
1560 POKE PMBASE+1536+K+T,A:POKE PMBAS
E+1792+K+T,A:NEXT T:NEXT K
3010 POKE 704,202:POKE 705,122:POKE 70
6,74:POKE 707,250:POKE 623,16:POKE 708
,108:POKE 709,172:POKE 710,150
3015 POKE 711,54
3020 DL=PEEK(560)+256*PEEK(561):FOR K=
8 TO 24 STEP 4:POKE DL+K,PEEK(DL+K)+12
8:NEXT K:POKE 756,CHSET/256
3030 FOR T=2 TO 22 STEP 4:POSITION 0,T
:? #6;"####################":NEXT T
3040 HI=INT(VBLANK/256):POKE 54286,0:P
OKE 547,HI:POKE 546,VBLANK-256*HI:HI=I
NT(DLI/256):POKE 513,HI
3050 POKE 512,DLI-256*HI:SC=0:HS=SC:LE
VEL=1:POKE 54281,CHSET/256
4000 REM WAIT FOR START/SELECT
```

```
4010 POP :POP :POSITION 1,3:? #6;"pres
s start/select"
4015 POSITION 6,5:? #6;"TRAFFIC":POSIT
ION 7,7:? #6;"panic"
4020 POSITION 0,0:? #6;"score:";SC
4022 POSITION 0,1:? #6;"high score:";H
S
4025 POSITION 12,0:? #6;"level";LEVEL;
"  "
4030 IF PEEK(53279)=6 THEN 7010
4035 IF PEEK(53264)=0 THEN 7010
4040 IF PEEK(53279)<>5 THEN 4030
4050 LEVEL=LEVEL*(LEVEL<9)+1:POSITION
17,0:? #6;LEVEL:POSITION 13,3:? #6;"SE
LECT"
4060 IF PEEK(53279)=5 THEN 4060
4070 POSITION 13,3:? #6;"select":GOTO
4030
7000 REM PRESSED START
7010 POSITION 7,3:? #6;"START"
7020 IF PEEK(53279)=6 THEN 7020
7030 POSITION 1,3:? #6;"
    ":SC=0:POSITION 6,0:? #6;SC;"    "
:POSITION 12,0:? #6;"      "
7110 STAGE=LEVEL:MEN=3:POSITION 6,5:?
#6;"       "
7115 POSITION 6,7:? #6;"STAGE ";STAGE:
POKE 77,0
7120 FOR K=0 TO 5:S(K)=8+STAGE*2.5+5-K
+128*(INT(K/2)=K/2)
7130 NEXT K
7135 FOR K=0 TO 5
7140 IF Y1(K)=0 THEN 7150
7145 POSITION X1(K),Y1(K):? #6;"#":FOR
 T=1 TO 3:IF Y1(K)+T>23 THEN 7148
7146 LOCATE X1(K),Y1(K)+T,A:IF A<39 TH
EN POSITION X1(K),Y1(K)+T:? #6;" "
7148 NEXT T
7150 IF Y2(K)=0 THEN 7160
7155 POSITION X2(K),Y2(K):? #6;"#":FOR
```

```
 T=1 TO 3:IF Y2(K)+T>23 THEN 7158
7156 LOCATE X2(K),Y2(K)+T,A:IF A<39 TH
EN POSITION X2(K),Y2(K)+T:? #6;" "
7158 NEXT T
7160 NEXT K
7170 FOR K=0 TO 5:N(K)=STAGE*(STAGE<5)
+4*(STAGE>4)-1:NEXT K
7190 FOR K=0 TO 23:POKE 1558+K,0:NEXT
K
7210 FOR T=0 TO 5:N=0:FOR K=0 TO N(T):
POKE 1558+T*4+K,N:N=N+25+INT(RND(0)*39
):NEXT K:NEXT T
7310 FOR K=0 TO 5:POKE 1552+K,S(K):NEX
T K
7320 POP :POP
7510 FOR L=0 TO 5:HPOS=56+8*INT(RND(0)
*18):GOSUB 20010
7520 POKE 1652+5-L,HPOS-1:X1(L)=X:Y1(L
)=Y:NEXT L
7530 FOR L=0 TO 5:POKE 1658+5-L,0:IF R
ND(0)<0.3 THEN HPOS=56+8*INT(RND(0)*18
):GOSUB 20010:POKE 1658+5-L,HPOS-1
7540 X2(L)=X:Y2(L)=Y:NEXT L
8030 FOR X=6 TO 13:LOCATE X,7,A:IF A>3
8 THEN A=32
8040 POSITION X,7:PUT #6,A:NEXT X
9030 FU=500:POSITION 12,0:? #6;"time "
;FU:POKE 207,50:POKE 208,50
10010 FOR T=PEEK(1669) TO PEEK(1669)+1
4:POKE PMBASE+768+T,0:NEXT T
10015 POKE 1668,100:POKE 1669,225
10020 POKE 1673,0:POKE 1665,0:POKE 166
7,0:POKE 1676,0
10030 POSITION 17,0:? #6;FU:COLOR 134
10060 FOR K=1 TO MEN:PLOT 16+K,1:NEXT
K
10070 IF MEN<3 THEN POSITION 17+MEN,1:
? #6;" ";
11010 POKE 1675,1:POKE 53278,0:POKE 54
```

```
286,192:A=USR(USER):SOUND 0,0,0,0:SOUN
D 1,0,0,0
11020 IF PEEK(53257)+PEEK(53258)+PEEK(
1676)>0 THEN 13010
12000 REM **END OF STAGE
12510 STAGE=STAGE+1:K=PEEK(88)+256*PEE
K(89):K=(PEEK(K+17)-16)*10+PEEK(K+18)-
16:K=K*10
12530 FOR FU=K TO 0 STEP -10:SC=SC+10*
STAGE:POSITION 6,0:? #6;SC:POSITION 17
,0:? #6;FU;:IF FU<100 THEN ? #6;" ";
12550 SOUND 0,70,10,10
12560 FOR K=0 TO 5:NEXT K:SOUND 0,0,0,
0:NEXT FU
12570 POSITION 17,0:? #6;" ";
12610 GOTO 7115
13000 REM **PLAYER DIED
13010 MEN=MEN-1:POSITION 17+MEN,1:? #6
;" "
13020 FOR T=0 TO 15:SOUND 0,255-T*17,1
0,10:POKE PMBASE+768+PEEK(1669)+T,0:FO
R K=0 TO 3:NEXT K:NEXT T
13100 IF MEN>0 THEN 10010
13110 POSITION 5,12:? #6;"game over":I
F SC>HS THEN HS=SC
13120 FOR K=1 TO 150:NEXT K:POSITION 5
,12:? #6;"         ":GOTO 4010
20000 REM
20010 X=(HPOS-56)/8:Y=2+(L*4)
20020 POSITION X,Y:? #6;"%":FOR T=1 TO
 3:IF Y+T<24 THEN POSITION X,Y+T:? #6;
"$"
20030 NEXT T
20040 RETURN
32000 GOTO 32000
```

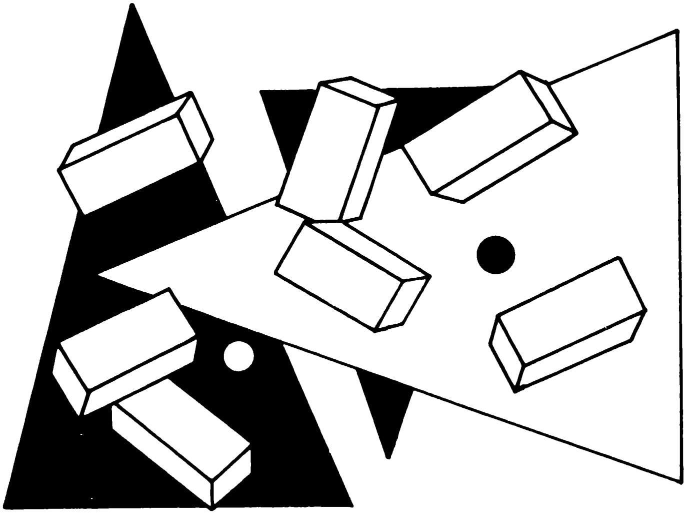
BREAKTHROUGH

(16K CASSETTE, 24K DISK)

This is the computer version of the old arcade game in which you chip away at a multi-coloured wall with your bat and ball. When the wall has been completely knocked out, your bat will decrease in size and a new wall will be drawn up. In this version the ball can bounce off the bat at 16 different angles.

Press SELECT to alter the number of balls you can miss before the game is over. Press OPTION to change to Breakthrough, when the ball, instead of bouncing off the wall, will wipe through it. Press 'f' or 's' to change to a fast or a slow ball. Press START to start the game.

Use a joystick in port 1 or move left and right, and press the fire button to slow down the movement of your bat.



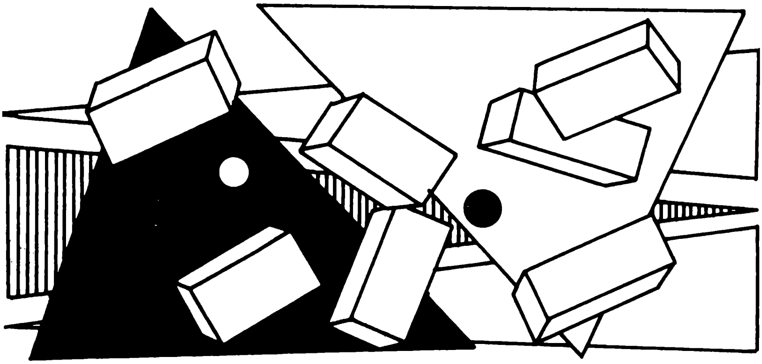
```
0 REM *** BREAKTHRU, BY G. RYAN
10 DIM A$(5):GOTO 50
20 RESTORE D:TRAP 40
30 FOR X=A TO A+400:READ D:POKE X,D:NE
XT X
40 POP :RETURN
50 RESTORE 50:READ N,N1,N2,N3,N4,N5,N6
,N7,N8,N9,N10,N11,N12,N16,N20,F,W;KB,C
SL:DATA 0,1,2,3,4,5,6
55 DATA 7,8,9,10,11,12,16,20,255,256,7
64,53279
59 REM SETUP TOP OF RAM
60 R=INT(PEEK(740)/N4)*N4:POKE 106,R:G
RAPHICS N20+N2:POKE 106,R-N8:RT=R*W
70 VBI=RT-180:VBD=RT-360:PLR=RT-512:MS
L=RT-640:MAIN=RT-960:DLI=RT-1080:PMB=R
-N4:INIT=RT-1024:DL=RT-1150
80 GRAPHICS N2+N16:GOSUB 4500:POSITION
 N6,N5:? #6;"breakthru":POSITION N1,N7
:? #N6;"WRITTEN BY G. RYAN"
90 D=5500:A=VBI:GOSUB N20:D=6500:A=VBD
:GOSUB N20
100 D=5000:A=MAIN:GOSUB N20:D=7500:A=D
LI:GOSUB N20
110 D=6000:A=1024:GOSUB N20:D=8000:A=1
760:GOSUB N20:D=7000:A=INIT:GOSUB N20
119 REM SELECT GAME SCREEN
120 GRAPHICS N2+N16:GOSUB 4500:SETCOLO
R N,N7,N6:SETCOLOR N1,N3,N8:SETCOLOR N
2,13,N10:SETCOLOR N3,N10,N12:G=N
130 T=N:? #N6;"  SCORE:";SC;:POKE 85,1
3:? #N6;"|HI:|";HS:POSITION N5,N3:? #N
6;"|breakthru|"
140 ? #N6:? #N6;" press option/select"

150 ? #N6;"    |NO. OF BALLS:| 3":B=N3:
L=N7:D=N:POKE 764,62:POSITION N8,N9:?
#N6;"BREAKOUT"
160 GOSUB 4000:C=PEEK(CSL):X=STRIG(N):
IF C=N5 OR C=N6 OR X=N THEN 190
```

```
170 D=D-0.2:D=D*(D>N):SOUND N,F,N10,D:
SOUND N1,F-N1,N10,D:IF D>N9 THEN 160
175 IF PEEK(KB)=58 THEN 3500
180 GOTO 160
190 IF C=N6 OR X=N THEN 230
200 IF D>N9 THEN 170
210 D=N10:B=B+N1:IF B>N5 THEN B=N1
220 POSITION 17,N7:? #N6;B:GOTO 160
230 POKE 53761,N:POKE 53763,N:B=B+N1:G
OSUB 4020:GOSUB 3550
240 POKE 205,N:POKE 203,MSL-INT(MSL/W)
*W:POKE 204,INT(MSL/W):GOSUB 4100
250 POKE 208,N:POKE 209,N6:POKE 21,120
:POKE 22,N1:POKE 24,76:POKE 25,44:POKE
 26,N:POKE 82,N:POKE 83,39
260 POKE 27,N1:POKE 1024,N1:POKE 1025,
N1:POKE 1026,N:POKE 1027,N16
270 POKE 1038,N2:POKE 1039,N2:POKE 104
6,B:POKE 1784,B+N16:POKE 1047,N:POKE 1
048,96
279 REM SETUP DLIST
280 GRAPHICS N4:POKE 559,N:GOSUB 4500:
COLOR N1:FOR A=N TO N3:PLOT N4+A,N:DRA
WTO N4+A,39:PLOT 72+A,N
290 SETCOLOR N,N7,N8:DRAWTO 72+A,39:NE
XT A:POKE DL,112:POKE DL+N1,112:POKE D
L+N2,112:SETCOLOR N1,N11,N8
300 PLOT N4,N:DRAWTO 72,N:POKE DL+N3,7
3:POKE DL+N4,PEEK(88):POKE DL+N5,PEEK(
89):SETCOLOR N3,N3,N8
310 FOR A=DL+N6 TO DL+N11:POKE A,N9:NE
XT A:POKE DL+N12,137:IF G=N3 THEN POKE
 1048,N
320 POKE DL+13,202:POKE DL+14,N:POKE D
L+15,N6:FOR A=DL+N16 TO DL+N16+N5:POKE
 A,138:NEXT A
330 POKE DL+21,201:A=PEEK(88)+PEEK(89)
*W+150:POKE DL+22,A-INT(A/W)*W:POKE DL
+23,INT(A/W)
340 FOR A=DL+24 TO DL+48:POKE A,N9:NEX
```

```
T A:POKE DL+49,70:POKE DL+50,234:POKE
DL+51,N6
350 POKE DL+52,70:POKE DL+53,PEEK(560)
:POKE DL+54,PEEK(561):POKE DL+55,N6:PO
KE DL+56,65
360 POKE DL+57,DL-INT(DL/W)*W:POKE DL+
58,INT(DL/W):POKE 560,PEEK(DL+57):POKE
 561,PEEK(DL+58):POKE 559,46
370 FOR A=1536 TO 1636 STEP N20:POKE A
,N:POKE A+19,N:POKE A+N1,85:POKE A+18,
85
380 FOR X=A+N2 TO A+N16+N1:POKE X,F:NE
XT X:NEXT A
400 POKE 54279,PMB:POKE 53277,N3:POKE
704,86:POKE 705,184:POKE 623,N4
405 IF G=N3 THEN POKE 1026,N1
410 POKE PLR+84,F:POKE PLR+85,F:POKE 5
3256,N1:POKE 53248,120:POKE 53260,N
420 POKE 656,N:POKE 658,N:POKE 657,25:
? "hi score ";HS;
425 IF G=N3 THEN POKE 657,24:? "  demo
 mode    "
430 POKE 54286,F:POKE 205,N1:X=USR(INI
T,DLI,VBI,VBD,MAIN):SOUND N,N,N,N:SOUN
D N1,N,N,N
440 POKE 205,N:POKE 656,N:POKE 658,N:P
OKE 657,N6:? "|game over| ";:POKE 657,
24:? " PRESS start   ";
450 IF PEEK(CSL)=N3 THEN POKE 657,N4:?
 "|game aborted|"
460 FOR X=N1 TO N4:A$(X,X)=CHR$(PEEK(1
776+X)+32):NEXT X:SC=VAL(A$(N1,N4)):IF
 SC>HS AND G<>N3 THEN HS=SC
470 IF PEEK(CSL)<>N6 THEN 470
480 FOR X=53248 TO 53255:POKE X,N:NEXT
 X:POKE 53277,N
490 GOTO 110
999 GOTO 999
3499 REM GAME OPTIONS
3500 RESTORE 3510:FOR A=VBD+21 TO VBD+
25:READ D:POKE A,D:NEXT A:POKE 1029,N:
POKE 1030,N:GOSUB 4020:G=N3
3510 DATA 173,10,210,41,7
3520 POKE VBD+158,21:SOUND N,N,N,N:SOU
ND N1,N,N,N:POKE 1029,N:POKE 1030,N:GO
TO 240
3550 RESTORE 3560:FOR A=VBD+21 TO VBD+
25:READ D:POKE A,D:NEXT A:POKE VBD+158
,60
3560 DATA 165,24,56,229,21,170
3570 RETURN
4000 X=PEEK(KB):POSITION N3,N9:IF X=56
 THEN ? #N6;"|FAST|":G=N1:GOTO 4070
4010 IF X=62 THEN ? #N6;"slow":G=N:GOT
O 4070
4015 GOTO 4070
4020 IF G=N1 THEN 4050
4030 POKE 1029,N2:POKE 1030,254:POKE 1
042,255:POKE 1043,N1
4040 GOTO 4070
4050 POKE 1029,N3:POKE 1030,253:POKE 1
042,254:POKE 1043,N2
4070 IF PEEK(CSL)<>N3 THEN RETURN
4080 T=(T=N):POSITION N8,N9:IF T=N THE
N ? #N6;"BREAKOUT   ":GOTO 4095
4090 IF T=N1 THEN ? #N6;"BREAKTHRU":GO
TO 4095
4095 FOR X=N1 TO 30:NEXT X:RETURN
4100 IF T=N THEN 4150
4110 POKE VBD+66,234:POKE VBD+67,234:P
OKE VBD+112,N:RETURN
4150 POKE VBD+66,208:POKE VBD+67,84:PO
KE VBD+112,N1:RETURN
4500 POKE N16,64:POKE 53774,64:RETURN

4999 REM MAIN ROUTINE
5000 DATA 173,24,4,208,126,133,205,169
,96,141,24,4,234,234,234,169,255,160,1
6,153,1,6,153,21,6
5005 DATA 153,41,6,153,61,6,153,81,6,1
```

```
53,101,6,136,208,235,133,22,200,132,27
,169,16,174,253,6
5010 DATA 232,224,26,176,5,142,253,6,2
08,19,141,253,6,174,252,6,232,224,26,1
76,5,142,252,6,208
5015 DATA 3,141,252,6,162,8,142,3,4,20
2,189,115,4,157,41,4,189,123,4,157,57,
4,202,208,241
5020 DATA 160,0,152,140,8,208,162,3,15
7,25,4,202,208,250,169,1,162,3,157,29,
4,202,208,250,169
5025 DATA 188,141,9,4,133,205,174,23,4
,240,71,169,16,172,244,6,200,192,26,17
6,5,140,244,6,208
5030 DATA 51,141,244,6,172,243,6,200,1
92,26,176,5,140,243,6,208,35,141,243,6
,172,242,6,200,192
5035 DATA 26,176,5,140,242,6,208,19,14
1,242,6,172,241,6,200,192,26,176,5,140
,241,6,208,3,141
5040 DATA 241,6,206,23,4,208,187,173,2
,4,208,54,133,205,160,127,145,203,136,
208,251,206,248,6,206
5045 DATA 22,4,240,59,169,1,133,27,169
,76,133,24,169,48,133,25,169,0,133,26,
169,4,141,14,4
5050 DATA 141,15,4,133,22,141,2,4,172,
132,2,208,251,200,132,205,173,31,208,4
1,4,208,3,133,205
5055 DATA 96,174,132,2,189,12,4,141,0,
4,108,95,4,96,-1
5499 REM IMMEDIATE VBLANK
5500 DATA 216,206,1,4,208,27,173,0,4,1
41,1,4,173,120,2,74,74,41,3,170,173,21
,0,24,125,4
5505 DATA 4,221,8,4,240,2,133,21,165,2
1,133,21,141,0,208,165,205,208,3,76,95
,228,206,16,4,208,32
5510 DATA 173,14,4,141,16,4,165,24,166
,26,24,125,18,4,221,20,4,240,4,133,24,
208,9,165,25
5515 DATA 73,1,133,26,32,97,4,164,25,1
69,0,145,203,200,145,203,165,24,141,5,
208,206,17,4,208
5520 DATA 20,173,15,4,141,17,4,165,25,
166,27,24,125,18,4,133,25,201,19,144,3
1,164,25,169,255
5525 DATA 145,203,200,145,203,192,92,1
44,5,169,0,141,2,4,206,114,4,208,5,162
,0,142,1,210,76
5530 DATA 95,228,162,1,134,27,202,134,
22,32,97,4,24,144,212,-1
5999 REM TABLES
6000 DATA 1,1,1,16,0,2,254,0,0,180,60,
0,2,1,2,2,2,2,255,1,64,190,3,0,96,0,0,
0,0,0,0,0,0,1,1,1,1,1,1,1,1
6005 DATA 1,1,1,1,1,1,1,2,2,1,1,1,1,1,
1,1,3,3,2,2,1,1,1,1,1,1,1,1,2,2,3,3,2,
22,42,62,82,102
6010 DATA 0,0,10,8,5,3,2,1,140,150,156
,163,134,183,179,167
6015 DATA 0,0,169,199,141,0,210,169,20
1,141,1,210,169,5,141,114,4,96,0,0
6020 DATA 1,1,2,3,3,2,1,1,2,1,1,1,1,1,
1,2,-1
6499 REM DEFFERRED VBLANK
6500 DATA 169,0,133,59,165,205,208,3,7
6,98,228,165,25,201,82,208,47,165,27,2
40,43,165,24,56,229
6505 DATA 21,170,236,3,4,176,32,189,41
,4,141,14,4,141,16,4,189,57,4,141,15,4
,141,17,4
6510 DATA 169,0,133,22,133,27,189,25,4
,133,26,32,97,4,165,22,208,84,165,25,2
01,44,176,78,201
6515 DATA 32,144,74,24,105,224,74,170,
189,73,4,133,208,134,0,165,24,24,105,1
92,74,74,74,168,177
6520 DATA 208,240,49,133,22,169,0,145,
208,165,27,73,1,133,27,173,10,210,41,7
```

```
,170,189,87,4,141
6525 DATA 0,210,169,167,141,1,210,169,
5,141,114,4,206,24,4,166,0,189,81,4,24
,109,23,4,141
6530 DATA 23,4,165,24,56,233,3,133,60,
76,98,228,-1
6999 REM INITIALIZING ROUTINE
7000 DATA 104,169,0,141,14,212,104,141
,1,2,104,141,0,2,104,141,35,2,104,141,
34,2,104,141,37
7005 DATA 2,104,141,36,2,104,141,96,4,
104,141,95,4,169,192,141,14,212,133,20
5,108,95,4,-1
7499 REM DLIST INTERRUPT
7500 DATA 72,152,72,164,59,192,7,208,4
,160,0,132,59,185,224,6,141,10,212,141
,24,208,200,132,59
7505 DATA 104,168,104,64,-1
7999 REM TABLES
8000 DATA 252,250,248,246,244,242,244,
0,0,0,0,243,227,239,242,229,218,16,16,
16,16,0,98,90,16,0,183,154,16,17
8005 DATA -1
```



SHOOTOUT

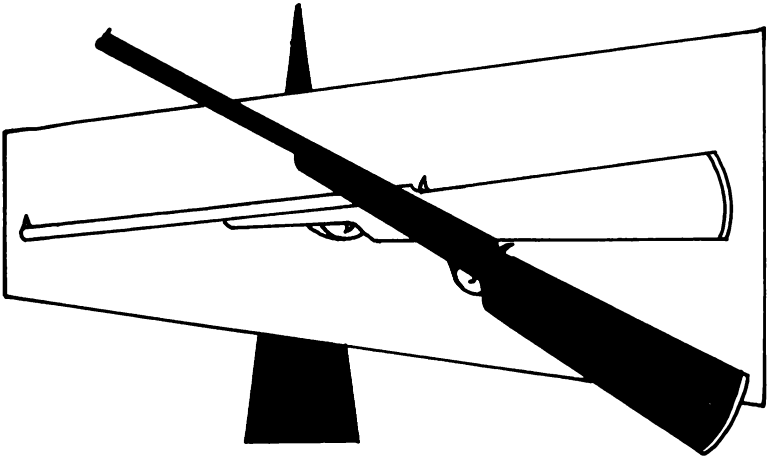
(16K CASSETTE, 24K DISK)

You and your opponent are neighbours. However, because of a feud between your families which started longer ago than anyone can remember, you are taking turns at shooting one another over the hill that separates your properties.

After running the program, the landscape is drawn up, the wind speed is shown, and the question 'angle?' will be printed at the bottom of the screen. The first player must then type in the angle he wishes to shoot at (from 0 to 90).

When the question 'bags?' is printed, the first player must type in the number of bags of gunpowder behind the shot (usually from 0 to 20). After this the cannonball will be shown shooting across the screen.

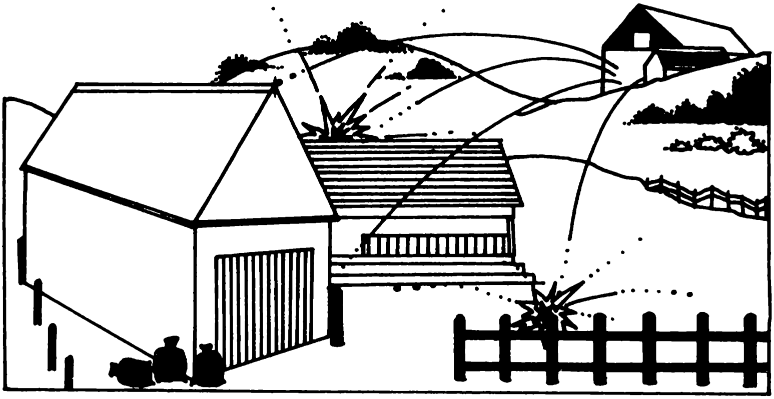
Then the second player has a turn. This is repeated until someone hits their opponent. The score for both players will then be shown and another landscape will be drawn.



```
5 REM ****  SHOOTOUT  ***
10 REM *** written by ***
15 REM ***  G. Ryan.  ***
20 GOTO 4900
30 T=N1:P=-P:GOSUB 2500
40 SOUND N,N,N,N:Y1=N1:TRAP 40:GOSUB 5
00
50 E=VAL(C$):Y1=N2:GOSUB 500:IF LEN(C$
)>4 THEN IF C$(1,5)="ABORT" THEN 5100
60 IN=VAL(C$)/N2:R=(E-90)*P/57.2958:XI
=IN*SIN(R):YI=IN*COS(R):GOSUB 300
70 X=X((P+N1)/N2):Y=H(X((P+N1)/N2))+N2

80 XT=X:YT=Y
90 IF Y<N THEN 200
100 IF X<N OR X>159 THEN 200
110 SOUND N,F*(F>N),N10,N2:F=F-YI:IF Y
>95 THEN 130
120 COLOR N2:PLOT X,96-Y:COLOR N:PLOT
XT,96-YT:XT=X:YT=Y
130 XI=XI+(WN-XI)*XA:YI=YI-G:X=X+XI:Y=
Y+YI
140 IF X<N OR X>159 THEN 200
150 IF Y>H(X) THEN 90
160 X=X+XI*(H(X)-Y)/YI
170 X=INT(X):IF X>159 THEN X=159
180 Y=H(X):GOTO 200
200 COLOR N:PLOT XT,96-YT:IF INT(X((N1
-P)/N2)/N2)<>INT(X/N2) THEN 30
210 S((P+N1)/N2)=S((P+N1)/N2)+N1:COLOR
 N2
220 FOR Z=-N6 TO N6 STEP N2:PLOT X,95-
Y:DRAWTO X+Z,90-Y:NEXT Z
230 FOR Z=15 TO N STEP -0.6:FOR A=N TO
 2:SOUND A,RND(0)*50+100,8,Z:NEXT A:SE
TCOLOR 4,3,Z-(Z>13)+RND(N)*2:NEXT Z
240 POKE 712,N:FOR A=N TO N2:SOUND A,N
,N,N:NEXT A
250 FOR A=N TO 1000:NEXT A:GOTO 5100
300 FOR A=15 TO N STEP -N2:SOUND N,RND
```

```
(N)*50+120,8,A:NEXT A:F=150
310 RETURN
500 POKE 764,255:POKE 752,N:POKE 657,X
P((P+N1)/N2):POKE 656,Y1:C$=""
510 INPUT C$:POKE 657,XP((P+N1)/N2):PO
KE 752,N1:POKE 656,Y1:? ":"
520 RETURN
2000 POKE 658,N:POKE 657,XP:POKE 656,Y
P
2010 RETURN
2500 COLOR N2:PLOT X(N),96-H(X(N)):DRA
WTO X(N),95-H(X(N))
2510 PLOT X(N1),96-H(X(N1)):DRAWTO X(N
1),95-H(X(N1))
2520 RETURN
4900 RESTORE 4900:DATA 0,1,2,3,6,10
4910 READ N,N1,N2,N3,N6,N10
5000 DIM H(159),X(N1),S(N1),C$(11),CL$
(N1),XP(N1)
5010 X(N)=9:X(N1)=150:S(N)=N:S(1)=N:CL
$=CHR$(125):XP(N)=7:XP(N1)=28:POKE 82,
N2:POKE 83,39
5020 GRAPHICS 23:DL=PEEK(560)+256*PEEK
(561)+N2:H=INT(DL/256):L=DL-H*256:POKE
 559,N:POKE 560,L:POKE 561,H
5030 FOR A=DL+75 TO DL+105:IF PEEK(A)<
>65 THEN NEXT A
5040 POP :A=A-5:POKE A,66:POKE A+N1,PE
EK(660):POKE A+N2,PEEK(661):POKE A+3,N
2:SETCOLOR N2,N,N
5050 POKE A+4,N2:POKE A+5,N2:POKE A+N6
,65:POKE A+7,L:POKE A+8,H
5060 POKE 559,34:POKE 659,N:POKE 703,4
:P=N1
5070 SETCOLOR N,15,N6:SETCOLOR N1,N3,1
2:SETCOLOR N2,15,N
5100 POKE 752,N1:? #N6;CL$:? CL$;:POKE
 657,N3:? S(N);:POKE 657,N10:? "SHOOTO
UT by G. Ryan";:POKE 657,36
5110 ? S(N1):RAD :MH=RND(N)*75+5:LE=IN
```

```
T(RND(N)*25)+8:RE=INT(RND(N)*25)+8
5120 MW=10*RND(N)+6:XH=85*RND(N)+38:WN
=16*RND(N)-8:G=0.1:XA=3.0E-03
5130 A=XH/MW:B=1/MW:C=LE/MH:D=RE/MH:CO
LOR N1
5140 FOR I=N TO 159:J=A-I*B:IF J*-J<-2
93 THEN J=17.1172427
5150 Y=MH*EXP(-J*J):H=LE+Y-Y*C:IF I>XH
 THEN H=RE+Y-Y*D
5160 H=H+INT(RND(N)*2.5)
5180 PLOT I,95:DRAWTO I,95-H:H(I)=H:NE
XT I:? CL$;
5190 XP=N2:YP=N1:GOSUB 2000:? "Angle:"
;:POKE 657,23:? "Angle:"
5200 XP=N3:YP=N2:GOSUB 2000:? "Bags:";
:POKE 657,24:? "Bags:"
5210 Z=INT(4*ABS(WN)+0.5):XP=17:YP=N:G
OSUB 2000:IF WN>N THEN ? "---";Z;"-->"
:GOTO 5230
5220 ? "<--";Z;"---"
5230 XP=4:YP=N:GOSUB 2000:? "PLR (1) "
;S(N);
5240 POKE 657,28:? "PLR (2) ";S(N1):PO
KE 752,N:GOTO 30
```



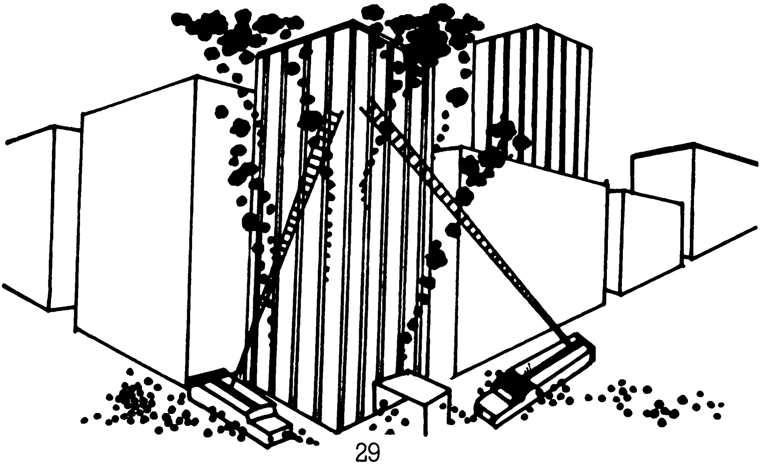
TOWER OF FLAMES

(16K CASSETTE, 24K DISK)

The skyscraper is ablaze and, in a frenzied, panic-stricken attempt to survive, its occupants are throwing themselves off the top of the building. As chief fire-fighter, you have been put in charge of saving them before they meet an untimely end on the hard pavement below. You must catch every falling individual in your net — a miss means certain death for a fellow human being, and three misses means you're out of a job.

POKE in the machine language and the computer will print up the title screen. Press SELECT to choose the stage you wish to start on, and press START to begin the game. Use a joystick in port 1 to move your net up, down, left and right.

Each man rescued gains you 10 points, multiplied by the stage you are on. After three misses the program returns to the title screen, displaying the high score.



```
10 REM **TOWER OF FLAMES
20 REM Written by C.McConnell,1984
100 MYPMBASE=0:PPOS=206:PHITE=207:DEDF
LAG=208:VFLAG=209
210 TOP=PEEK(740):PMBASE=(TOP-8)*256:C
HBASE=(TOP-12)*256:START=(TOP-10)*256
310 IVBLANK=START:USER=START+530
360 POKE 105,TOP:GRAPHICS 0:POKE 106,T
OP-3:GRAPHICS 0
410 POKE 106,TOP-5:GRAPHICS 18
430 POSITION 2,2:? #6;"TOWER OF FLAMES
"
450 POSITION 4,4:? #6;"please wait"
460 GOTO 810
500 REM ML DATA
510 DATA 165,209,240,3,76,95,228,173,4
8,6,74,74,74,74,141,80,6,173,48,6,41,1
5,24,109,49,6
511 DATA 141,49,6,201,16,144,8,41,15,1
41,49,6,238,80,6,162,3,189,50,6,240,56
,189,62,6,221
512 DATA 58,6,176,17,169,0,157,50,6,16
9,1,157,54,6,169,2,157,70,6,208,71,189
,62,6,56,237
513 DATA 80,6,157,62,6,222,74,6,208,56
,169,4,157,74,6,189,70,6,73,1,157,70,6
,24,144,40
514 DATA 189,54,6,240,53,165,206,56,23
3,8,221,62,6,176,107,24,105,16,221,62,
6,144,99,165,207,221
515 DATA 66,6,176,92,24,105,4,221,66,6
,144,84,176,6,24,144,103,24,144,149,16
9,0,157,54,6,188
516 DATA 66,6,169,15,208,2,240,88,141,
81,6,138,24,101,0,24,105,4,133,205,169
,0,133,204,145,204
517 DATA 136,206,81,6,208,248,173,84,6
,160,8,24,113,88,201,26,144,14,56,176,
2,16,198,233,10,145
518 DATA 88,169,1,136,208,235,145,88,2
```

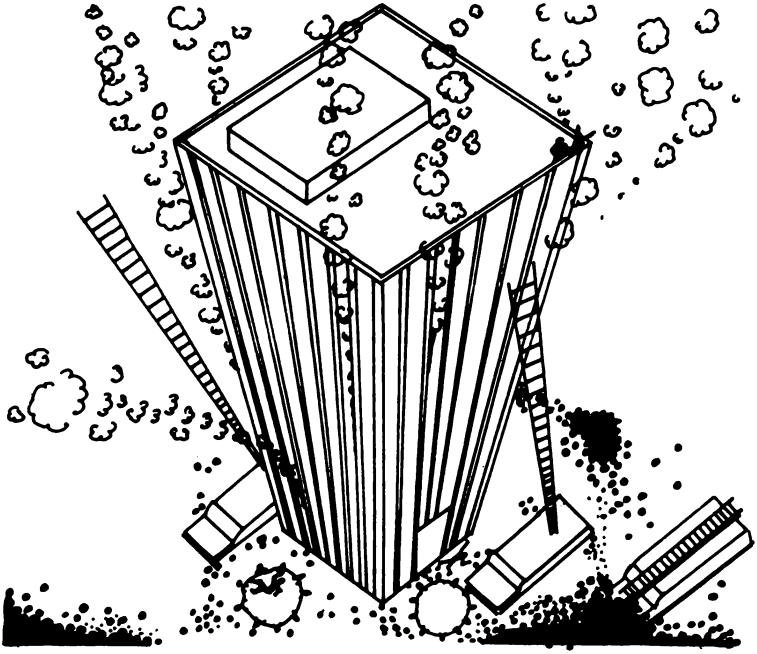
```
06,82,6,208,78,230,209,76,95,228,189,6
6,6,201,220,144,9,169
519 DATA 1,133,209,133,208,76,95,228,2
4,109,80,6,157,66,6,24,144,47,206,85,6
,208,42,173,83,6
520 DATA 141,85,6,169,1,157,50,6,169,0
,157,54,6,173,10,210,41,127,24,105,50,
157,58,6,169,63
521 DATA 157,66,6,169,220,157,62,6,169
,0,157,70,6,202,16,157,173,0,211,106,1
76,4,198,207,198,207
522 DATA 105,176,4,230,207,230,207,166
,207,224,206,144,4,160,206,132,207,224
,158,176,4,160,158,132,207,106
523 DATA 176,20,174,86,6,198,206,202,2
08,251,165,206,201,48,176,27,169,48,13
3,206,208,21,106,176,18,174
524 DATA 86,6,230,206,202,208,251,165,
206,201,210,144,4,169,210,133,206,165,
206,141,4,208,24,105,2,141
525 DATA 5,208,24,105,2,141,6,208,24,1
05,2,141,7,208,165,0,24,105,7,133,205,
169,0,133,204,162
526 DATA 3,189,50,6,24,125,54,6,240,41
,138,72,188,66,6,189,70,6,10,10,10,10,
24,105,15,170
527 DATA 169,16,141,81,6,189,0,6,145,2
04,136,202,206,81,6,208,244,104,170,18
9,62,6,157,0,208,198
528 DATA 205,202,16,201,169,0,164,207,
136,136,145,204,200,145,204,200,169,25
5,145,204,200,145,204,200,169,0
529 DATA 145,204,200,145,204,206,92,6,
208,23,173,93,6,141,92,6,172,10,210,17
7,88,201,130,240,4,201
530 DATA 131,208,4,73,1,145,88,76,95,2
28,169,0,133,209,104,165,209,240,252,9
6,-1
600 REM CHSET DATA
```

```
610 DATA 255,129,129,129,129,129,129,2
55
620 DATA 64,98,243,247,247,255,127,126
630 DATA 0,32,114,119,119,127,127,62
640 DATA -1
700 REM PAGE 6 DATA
710 DATA 0,0,0,24,24,24,24,8,8,56,8,24
,56,40,40,64
720 DATA 0,0,0,24,24,24,24,8,8,24,24,2
4,24,8,8,24
730 DATA 0,0,0,24,153,153,90,66,60,60,
24,60,36,36,36,36
800 REM **READ IN ML
810 TRAP 2030
2010 FOR K=START TO START+9999:READ A:
POKE K,A:NEXT K
2020 REM **READ IN CHARACTER SET
2030 POP :TRAP 3030
2040 FOR T=0 TO 511:POKE CHBASE+T,PEEK
(57344+T):NEXT T
3010 FOR K=CHBASE+8 TO CHBASE+9999:REA
D A:POKE K,A:NEXT K
3020 REM **READ IN PAGE 6 DATA
3030 POKE 623,17:POKE 756,CHBASE/256:P
OP :TRAP 4030
4010 FOR K=1536 TO 1793:READ A:POKE K,
A:NEXT K
4030 POP
4510 LEVEL=1
4710 POKE 54279,TOP-8:POKE 53277,3:POK
E 559,62:POKE MYPMBASE,TOP-8:HS=0:SC=0
5020 POKE 708,6:POKE 709,72:POKE 710,5
2:POKE 711,78
5050 ? #6;CHR$(125)
5060 POSITION 0,0:? #6;"score";SC
5070 POSITION 0,1:? #6;"high score ";H
S
5090 POSITION 13,0:? #6;"level";LEVEL;
" "
```

```
5110 POSITION 0,3:? #6;"press start/se
lect"
5510 K=PEEK(53279):T=PEEK(53264)
5600 IF K=6 OR T=0 THEN 6110
5700 IF K=5 THEN LEVEL=LEVEL*(LEVEL<9)
+1:POSITION 18,0:? #6;LEVEL;" "
5710 IF PEEK(53279)=K THEN 5710
5800 GOTO 5510
6000 REM **START GAME
6110 POKE VFLAG,1:POKE 54286,0
6140 HI=INT(IVBLANK/256):POKE 547,HI:P
OKE 546,IVBLANK-HI*256
6180 POKE 54286,64
7010 STAGE=LEVEL:SC=0:MEN=3:POSITION 5
,0:? #6;"00000"
7610 POSITION 0,1:? #6;"
  "

7710 POSITION 0,3:? #6;"
    "

7740 POSITION 13,0:? #6;"left:";MEN
8000 REM **SET DIFFICULTIES
8110 K=STAGE:IF K>10 THEN K=10
8120 POKE 1620,K:K=100-STAGE*2:IF K<50
 THEN K=50
8130 POKE 1619,K:K=14+STAGE*2
8140 POKE 1584,K:N=K/4:IF N>10 THEN N=
10
8150 POKE 1622,N
8210 POKE 1618,6+STAGE*2
8310 POKE 1628,1:POKE 1629,1
8500 REM **SETUP SCREEN
8510 COLOR 33:FOR T=2 TO 10:PLOT 0,T:D
RAWTO 19,T:NEXT T
8530 COLOR 95:PLOT 0,11:DRAWTO 19,11
8610 COLOR 162:FOR T=0 TO 30:X=RND(0)*
18:Y=RND(0)*8+2:PLOT X,Y:NEXT T
8920 POSITION 6,5:? #6;"STAGE ";STAGE
8940 SOUND 0,100,10,10:FOR T=0 TO 70:N
EXT T:SOUND 0,200,10,10:FOR T=0 TO 70:
NEXT T
```

```
8960 SOUND 0,150,10,10
9000 REM **SET ML VARIABLES
9010 FOR T=0 TO 1:POKE PMBASE+768+PEEK
(PHITE)+T,0:NEXT T
9030 POKE PHITE,200:POKE PPOS,100
9110 FOR T=0 TO 3:FOR K=0 TO 15:POKE P
MBASE+1024+T*256+PEEK(1602+T)-15+K,0:N
EXT K:NEXT T
9210 FOR T=1586 TO 1609:POKE T,0:NEXT
T
9260 FOR T=1610 TO 1613:POKE T,1:NEXT
T
9500 REM **SET BASIC VARIABLES(FUEL..)

9700 COLOR 33:PLOT 6,5:DRAWTO 13,5
9710 REM **ZERO PLAYER
9750 SOUND 0,0,0,0
10000 REM **HERE WE GO
10010 POKE 704,16*INT(16*RND(0))+14
10020 POKE 705,16*INT(16*RND(0))+14
10030 POKE 706,16*INT(16*RND(0))+14
10035 POKE 707,16*INT(16*RND(0))+14
10040 POKE DEDFLAG,0:A=USR(USER):SOUND
 0,0,0,0:SOUND 1,0,0,0
11010 IF PEEK(DEDFLAG) THEN 15010
12000 REM **END OF STAGE
12010 K=0
12110 SC=SC+K:POSITION 5,0:? #6;SC
12210 STAGE=STAGE+1
12310 GOTO 8110
15000 REM **PLAYER DIED
15010 MEN=MEN-1
16000 REM **EXPLOSION
16010 FOR K=0 TO 4:IF PEEK(1602+K)<220
 THEN NEXT K
16020 POP
16030 FOR T=0 TO 15:FOR G=15 TO T STEP
 -1
16035 SOUND 0,T*16+G,10,10
16040 X=PMBASE+1024+256*K+205+G:POKE X
```

```
,PEEK(X-1):NEXT G:NEXT T
17000 REM **DISPLAY NUMBER OF MEN LEFT

17110 POSITION 13,0:? #6;"LEFT:";MEN
17210 SOUND 0,0,0,0
18000 IF MEN>0 THEN 9010
19000 REM **GAME OVER
19010 POSITION 5,6:? #6;"game over":FO
R K=1 TO 150:NEXT K
19020 COLOR 33:PLOT 5,6:DRAWTO 13,6
19030 SCRN=PEEK(88)+256*PEEK(89)+5:SC=
(PEEK(SCRN)-16)*1000+(PEEK(SCRN+1)-16)
*100+(PEEK(SCRN+2)-16)*10
19040 SC=(SC+(PEEK(SCRN+3)-16))*10
19050 IF SC>HS THEN HS=SC
19060 GOTO 5070
```



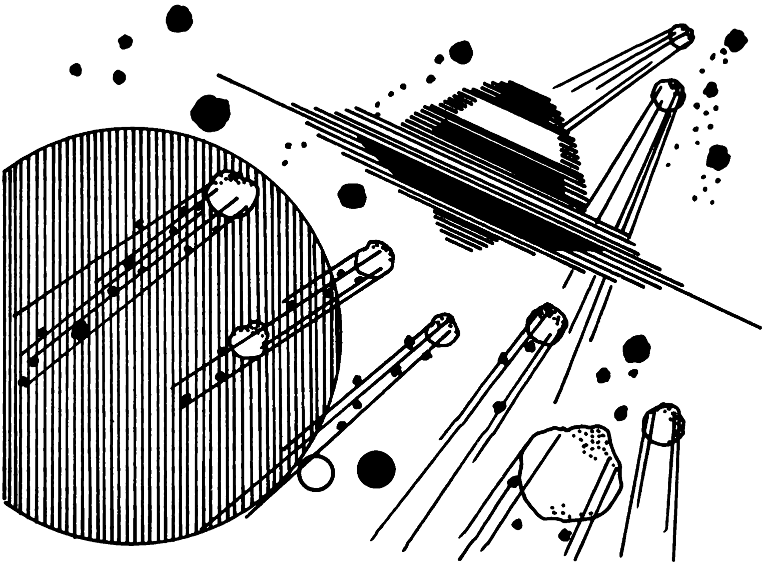
ASTEROID STORM

(16K CASSETTE, 16K DISK)

On your way to Alpha Centauri, you have been trapped in an asteroid storm. You must navigate your way through these huge moving rocks to the other side of the storm without being hit.

POKE in the machine language, and the computer will then display the title screen. Press START to begin the game. Use a joystick in port 1 to move left and right. Press the fire button to go forward faster.

Hitting pods situated among the rocks will give you bonus points. Points are also awarded if you make it to the other side. Every 10,000 points, an extra ship is awarded.



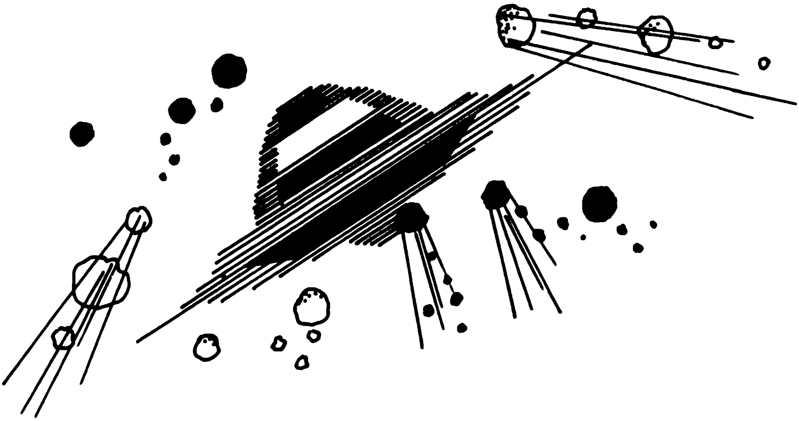
```
10 REM ***ASTEROID   STORM***
20 REM **BY CLIFF MCCONNELL**
50 REM **VARIABLES
60 N0=0:N1=1:N2=2:N3=3:N6=6:DIM H(N3),
I(N3):I(N0)=N3:I(N1)=N6+N6:I(N2)=48:I(
N3)=192
65 FOR G=N0 TO N3:H(G)=N0:NEXT G
70 DELAY=14552:WAIT=14553:VFLAG=14554:
DFLAG=14555:AFLAG=14556:BFLAG=14557:DL
I=14376:USER=14399:START=14080
80 M=14336:BASE=203:PPOS=205:PHITE=206
:SDELAY=208:SWAIT=209:HFLAG=14558
100 REM **SETUP GRAPHICS
110 POKE VFLAG,N1:POKE 106,59:GRAPHICS
 18:POKE BASE+N1,M/256+N2+N2:POKE BASE
,N0:POKE 623,16:HS=N0
115 POSITION N3,N3:? #N6;"ASTEROID STO
RM":POSITION N3+N1,N6:? #N6;"PLEASE W
AIT":POKE 53774,N0
120 DLIST=PEEK(560)+256*PEEK(561):POKE
 DLIST+N3,PEEK(DLIST+N3)+128
130 FOR K=N6 TO 16:POKE DLIST+K,PEEK(D
LIST+K)+128:NEXT K
140 POKE SWAIT,N6:POKE 53277,N3:POKE 5
59,62:POKE 54279,56:POKE PHITE,240:POK
E PPOS,100
145 FOR G=N0 TO 255:POKE M+768+G,N0:PO
KE M+1280+G,N0:POKE M+1536+G,N0:POKE M
+1792+G,N0:NEXT G
150 FOR K=N1 TO 12:RESTORE :FOR L=N1 T
O 14:READ A:POKE M+1297+L+K*16,A:POKE
M+1553+L+K*16,A
160 NEXT L:NEXT K
180 POKE 705,N0:POKE 706,N1:POKE 704,1
16:POKE 707,116
200 REM **ASTEROIDS
210 DATA 0,28,22,50,98,195,129,193,67,
70,100,36,60,0
300 REM **PLAYERS SHIP
310 DATA 8,28,28,62,62,62,127,0,0,0,0,
0,0,0,0,0
320 FOR K=N0 TO 15:READ A:POKE 14536+K
,A:NEXT K

500 REM **POKE IN MACHINE LANGUAGE
550 N=N0:RESTORE 32000
560 READ A:IF A<>-N1 THEN POKE START+N
,A:N=N+N1:GOTO 560
600 REM **SETUP INTERUPTS
620 A=USR(START):A=INT(DLI/256):POKE 5
13,A:POKE 512,DLI-256*A:POKE 54286,192

630 FOR G=N0 TO 13:POKE 705,G+240:POKE
 706,G:FOR K=N0 TO 15:NEXT K:NEXT G
650 ? #N6;CHR$(125);:POSITION N1,N6:?
#N6;"PLEASE PRESS START":IF SC>HS THEN
 HS=SC
660 POSITION N1,N0:? #N6;"score ";SC:P
OSITION N1,N1:? #N6;"high score ";HS:N
E=1
670 IF PEEK(53279)<>N6 THEN 670
690 ? #N6;CHR$(125):SC=N0:STAGE=N1:POS
ITION N1,N0:? #6;"score ";SC:L=N3:GOSU
B 12010
700 REM **SET ASTEROID SPEEDS
710 POKE WAIT,N2:POKE DELAY,N1:POKE VF
LAG,N1:POSITION N6,N6:? #N6;"STAGE ";S
TAGE;:POKE HFLAG,N0:BS=N1
715 RESTORE 13030:GOSUB 13010
717 GOSUB 10010
720 FOR K=N0 TO 11
730 X1=INT(RND(N0)*STAGE*4)+INT(RND(N0
)+0.5)*128+N1:X2=INT(RND(N0)*STAGE*4)+
INT(RND(N0)+0.5)*128+N1
740 POKE 14464+K,X1:POKE 14500+K,X2:NE
XT K
750 POKE VFLAG,N0:POKE PHITE,240
1000 REM **GAME STARTS
1010 POKE SDELAY,60:POKE AFLAG,N0
1030 POSITION N6,N6:? #N6;"            "
```

```
1040 POKE 53278,N0:POKE DFLAG,N0:POKE
AFLAG,N0:POKE VFLAG,N0
1100 A=USR(USER):ON PEEK(BFLAG) GOTO 1
510,1410,1220
1200 REM **MADE IT!
1220 GOSUB 11010
1250 STAGE=STAGE+N1:GOTO 710
1400 REM **HIT BONUS POD
1410 N=N0
1420 IF PEEK(53256+N)/N2<>INT(PEEK(532
56+N)/N2) THEN 1450
1430 N=N+N1:IF N<N2+N2 THEN 1420
1440 POKE DELAY,N1+N1:GOTO 1040
1450 H=(PEEK(PPOS)-40)/8:I=(PEEK(PHITE
)-52)/16
1455 K=100*INT(2*BS+0.5):POSITION H,I:
? #N6;K:SC=SC+K:BS=BS+N1
1460 FOR K=1200 TO N0 STEP -20:SOUND N
0,K,10,N2+N2:NEXT K
1470 FOR K=N0 TO N3:G=M+768+H(N)+K:POK
E G,PEEK(G)-I(N):NEXT K
1480 POSITION H,I:? #N6;"      ":POSIT
ION 7,N0:? #N6;SC
1490 SOUND N1,N0,N0,N0:GOSUB 3000
1495 GOTO 1440
1500 REM **HIT ASTEROID
1510 L=L-N1:GOSUB 2010:IF L<>N0 THEN 1
520
1515 POKE 53278,N0:POSITION N2+N3,N6:?
 #N6;"GAME OVER":FOR G=N0 TO 150:NEXT
G:GOTO 650
1520 N=PEEK(PHITE)
1530 POKE SDELAY,N0:POKE DELAY,N0:POKE
 PHITE,35+16*L:POKE PPOS,51:POKE HFLAG
,N1:POKE VFLAG,N0:G=PEEK(20)
1550 IF PEEK(20)=G THEN 1550
1560 POKE VFLAG,N1:GOSUB 12010:POKE DF
LAG,N1:POKE HFLAG,N0:POKE VFLAG,N0
1570 POKE SDELAY,N0:POKE DELAY,N0:IF P
EEK(PHITE)<N THEN 1570
```

```
1580 GOSUB 11010
1590 POKE AFLAG,N0:GOTO 700
2000 REM **EXPLOSION
2010 POKE AFLAG,N1:POKE VFLAG,N0:POKE
HFLAG,N1:POKE SDELAY,N0:POKE DELAY,N0
2020 FOR G=15 TO 8 STEP -N1:FOR K=N1 T
O G:SOUND N0,255-((G-8)*32)-K,10,10
2030 POKE 14536+K-N1,PEEK(14536+K):NEX
T K:NEXT G:POKE VFLAG,N1:POKE PHITE,PE
EK(PHITE)-N6-N2
2040 DATA 0,73,73,42,42,28,28,119,119,
28,28,42,42,73,73,0
2050 RESTORE 2040:FOR G=N0 TO 15:READ
A:POKE 14536+G,A:NEXT G:POKE VFLAG,N0:
G=PEEK(20)
2060 IF G=PEEK(20) THEN 2060
2070 POKE VFLAG,N1:FOR G=200 TO N0 STE
P -10:SOUND N0,G,10,10:NEXT G
2080 RESTORE 310:FOR G=N0 TO 15:READ A
:POKE 14536+G,A:NEXT G
2090 RETURN
3000 REM **EXTRA
3010 POKE VFLAG,N1:IF SC<NE*10000 THEN
 3050
3020 NE=NE+1:IF L>10 THEN 3050
3030 FOR G=N1 TO N3+N1:SOUND N0,25,10,
12:FOR K=N1 TO 10:NEXT K:SOUND N0,N0,N
0,N0:FOR K=N1 TO 40:NEXT K:NEXT G
3040 L=L+1:GOSUB 12010
3050 POKE VFLAG,N0:RETURN
10000 REM **PUT BONUS PODS ON SCREEN
10010 FOR G=N0 TO N3:FOR K=N0 TO N3:PO
KE M+768+H(G)+K,N0:NEXT K:NEXT G
10030 FOR G=N0 TO N3:H(G)=INT(RND(0)*1
55)+45:H=INT(RND(0)*151)+45
10040 FOR K=N0 TO N3:N=M+768+H(G)+K:PO
KE N,PEEK(N)+I(G):NEXT K
10050 POKE 53252+G,H:NEXT G
10060 RETURN
11010 POKE DFLAG,N1:POKE AFLAG,N1:POKE
```

```
 VFLAG,N0
11020 IF PEEK(PHITE)>239 THEN 11050
11030 SC=SC+10*STAGE:POSITION 7,N0:? #
N6;SC;:GOSUB 3000
11040 SOUND N0,50,10,10:FOR K=N1 TO N2
+N3:NEXT K:SOUND N0,N0,N0,N0:POKE DELA
Y,N0:GOTO 11020
11050 POKE DFLAG,N0:RETURN
12000 REM **DISPLAY SHIPS
12010 FOR G=N1 TO L-N1:RESTORE 310:FOR
 K=N0 TO 15:READ A:POKE M+1792+20+K+16
*G,A:NEXT K:NEXT G
12020 FOR K=N0 TO 15:POKE M+1792+20+K+
16*L,N0:NEXT K
12030 POKE 53251,51:RETURN
13000 REM **TUNE
13010 READ A,B:IF A<>-N1 THEN SOUND N0
,A,10,10:FOR G=N1 TO B*25:NEXT G:GOTO
13010
13020 SOUND N0,N0,N0,N0:RETURN
13030 DATA 121,2,0,.1,121,.33,108,.33,
96,.33,91,2,-1,-1
32000 DATA 162,55,160,11,169,6,32,92,2
28,104,96,173,218,56,240,3,76,224,55,1
73,220,56,208,63,162,0
32001 DATA 189,128,56,133,62,189,140,5
6,133,4,189,152,56,133,0,32,231,55,165
,4,157,140,56,165,0,157
32002 DATA 152,55,189,164,56,133,62,18
9,176,56,133,4,189,188,56,133,0,32,231
,55,165,4,157,176,56,165
32003 DATA 0,157,188,56,232,224,12,208
,195,173,222,56,208,18,173,0,211,106,1
06,106,176,3,76,105,55,230
32004 DATA 205,106,144,2,198,205,173,2
19,56,240,4,230,206,208,49,173,16,208,
208,26,206,216,56,208,29,173
32005 DATA 217,56,141,216,56,198,206,1
69,132,141,1,210,169,150,141,0,210,76,
158,55,169,0,141,1,210,141
```

```
32006 DATA 0,210,198,208,208,6,165,209
,133,208,198,206,165,205,201,200,144,4
,169,200,133,205,165,205,201,45
32007 DATA 176,4,169,45,133,205,165,20
6,201,40,176,4,169,40,133,206,160,0,15
2,145,203,136,208,251,164,205
32008 DATA 162,15,189,200,56,145,203,1
36,202,16,247,165,205,141,0,208,169,0,
133,207,76,95,228,165,62,41
32009 DATA 112,74,74,74,74,133,1,165,6
2,41,15,240,15,24,101,4,133,4,201,16,1
44,6,41,15,133,4
32010 DATA 230,1,165,62,41,128,208,14,
165,0,56,229,1,201,35,176,2,169,210,13
3,0,96,165,0,24,101
32011 DATA 1,201,210,144,2,169,35,133,
0,96,72,138,72,156,207,189,152,56,141,
1,208,189,188,56,141,2
32012 DATA 208,230,207,104,170,104,64,
104,173,12,208,240,4,169,1,208,29,173,
8,208,24,109,9,208,109,10
32013 DATA 208,109,11,208,41,1,240,4,1
69,2,208,8,165,206,201,40,208,220,169,
3,238,218,56,141,221,56,96,-1
```

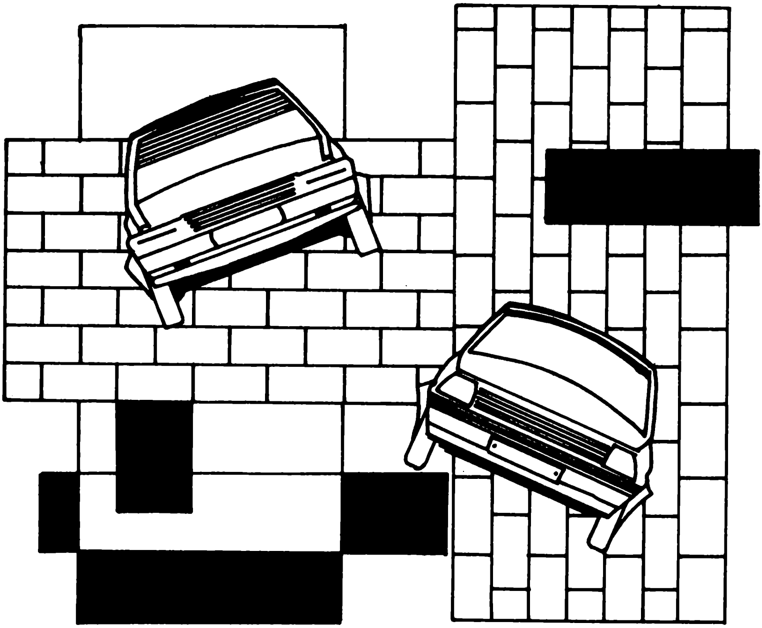


BLOCKOUT

(16K CASSETTE, 24K DISK)

It is the year 2109 and the differences between nations are being fought out in gladiatorial-type games, instead of in war. You and your opponent control 'death cars' which leave a deadly electrified wall behind them as they go. You can only move forwards, so the safe space left in the arena gets smaller and smaller. Outsmart your opponent by blocking him in.

After running the program, SELECT the number of games to be played before the winner can be chosen. Press OPTION for a one or two player game, and press START to begin. Use joysticks in port 1 and 2 to move left, right, up and down.



```
0 REM ***  BLOCKOUT    **
2 REM *** written by   **
5 REM *** G. Ryan,1984 **
10 GOTO 550
20 S=STICK(P):B=B+1:IF S<>15 THEN H(P)
=HI(S):V(P)=VI(S)
30 X=X(P)+H(P):Y=Y(P)+V(P):LOCATE X,Y,
C:COLOR P+1:PLOT X,Y:X(P)=X:Y(P)=Y:P=P
L(P):IF  NOT C THEN 20
40 FOR A=15 TO N STEP -0.7:SOUND N,PEE
K(53770),8,A:NEXT A:IF P=PL(P) THEN 60
50 PL(P)=P:B1=B:GOTO 20
60 B=B+B*PL-B1:B=INT(B/20)*20:S(P)=S(P
)+B:POKE 657,6:? "|bonus|";
70 FOR A=B TO N STEP -20:POKE 656,3:PO
KE 657,12:? A;"   ";:POKE 656,N1:POKE 6
57,4+10*P:? S(P)-A
80 SOUND N,80,10,10:FOR X=N1 TO 10:NEX
T X:SOUND N,N,N,N:FOR X=N1 TO 5:NEXT X
:NEXT A
90 IF S(N)>HS THEN HS=S(N)
100 IF S(N1)>HS THEN HS=S(N1)
110 FOR A=N1 TO 300:NEXT A:GOTO 810
500 POKE 82,N1:GRAPHICS 5:POKE 559,N:D
=PEEK(560)+256*PEEK(561):POKE 659,N1
510 FOR D=D+38 TO D+48:IF PEEK(D)<>66
THEN NEXT D
520 POP :POKE D,70:POKE D+3,6:POKE D+4
,6:POKE D+5,6:POKE 559,34
530 SETCOLOR 0,7,8:SETCOLOR 1,12,10
540 SETCOLOR 2,3,6:SETCOLOR 3,1,8:RETU
RN
550 DIM S(1),X(1),Y(1),H(1),V(1),RN(7)
,PL(1),HI(15),VI(15):N=0:N1=1:W=6:GOSU
B 500
560 S(N)=N:S(N1)=N:RESTORE 570:FOR A=N
 TO 7:READ D:RN(A)=D:NEXT A:CSL=53279
570 RESTORE 580:FOR A=N TO 15:READ X,Y
:HI(A)=X:VI(A)=Y:NEXT A:DATA 5,10,15,2
0,25,50,100,200
```

```
580 DATA 0,0,0,0,0,0,0,0,0,0,1,0,1,0,1
,0,0,0,-1,0,-1,0,-1,0,0,0,0,1,0,-1,0,0
600 ? #6;CHR$(125):COLOR 2:PLOT 0,0:DR
AWTO 79,0:DRAWTO 79,39:DRAWTO 0,39:DRA
WTO 0,0
610 RESTORE 1000:Z=N:COLOR 2
620 FOR A=13 TO 21
630 READ D:IF SGN(D)<>-1 THEN PLOT 14+
D,A:READ D:DRAWTO 15+D,A:GOTO 630
640 IF D=-3 THEN Z=Z+1:IF Z<3 THEN RES
TORE 1010:NEXT A
650 NEXT A:Z=0
660 ? CHR$(125):? "WRITTEN BY G. RYAN"
670 ? "|PLEASE PRESS START|":? "    |hi
 score:|";HS
680 POKE 709,Z:Z=Z*(Z<254)+2:SOUND 0,2
56-Z/3,10,3:SOUND 1,255-Z/3,10,3:IF PE
EK(CSL)<>6 AND STRIG(N) THEN 680
690 SOUND N,N,N,N:SOUND N1,N,N,N:IF ST
RIG(N)=N OR PEEK(CSL)=6 THEN 690
700 GRAPHICS 18:POSITION 6,2:? #W;"|bl
ockout|":POSITION 1,5
710 ? #W;"option : 1 PLAYER":POSITION
1,7:? #W;"select : 5 ROUNDS":FOR A=1 T
O 40:NEXT A:D=N
720 D=D-N1:A=PEEK(CSL):D=D+(A<>7):IF D
<1 THEN GOSUB 740:D=20
730 GOTO 720
740 IF A=6 OR STRIG(N)=N THEN RN=RN(RN
)+1:GOSUB 500:GOTO 810
750 IF A=5 THEN 800
760 IF A<>3 THEN 720
770 PL=PL=0:POSITION 10,5:IF PL=1 THEN
 ? #W;"2 PLAYERS":RETURN
780 ? #W;"1 PLAYER ":RETURN
800 RN=RN+1:RN=RN*(RN<8):POSITION 10,7
:? #W;RN(RN);" ROUNDS  ":RETURN
810 PL(N)=1:PL(1)=N:IF PL=N THEN PL(N)
=N
```

```
850 POP :? #W;CHR$(125):COLOR 3:PLOT 0
,0:DRAWTO 79,0:DRAWTO 79,39:DRAWTO 0,3
9:DRAWTO 0,0
860 RN=RN-1:? CHR$(125):? "P1:";S(0);:
POKE 657,11:? "p";CHR$(18);CHR$(26);S(
1):P=0:H=0:B1=0:B=0
870 ? "|ST:|";RN;:POKE 657,11:? "|HI:|
";HS
880 X(0)=9:X(1)=70:Y(0)=19:Y(1)=19:H(0
)=1:H(1)=-1:V(0)=0:V(1)=0
900 IF  NOT RN THEN 930
910 IF STRIG(0) THEN 910
920 GOTO 20
930 ? "|GAME OVER-hit start|"
940 IF PEEK(53279)<>6 THEN 940
950 GOTO 560
1000 DATA 0,5,-1
1010 DATA 0,0,5,5,9,9,26,26,48,48,-3,0
,6,9,9,12,15,20,23,26,26,29,29,32,35,3
9,39,43,43,46,50,-1
1020 DATA 0,0,6,6,9,9,12,12,16,16,19,1
9,26,26,28,28,32,32,36,36,39,39,43,43,
48,48,-1
1030 DATA 0,0,6,6,9,9,12,12,16,16,19,1
9,26,27,32,32,36,36,39,39,43,43,48,48,
-1
1040 DATA 0,0,6,6,9,9,12,12,16,16,19,1
9,26,26,28,28,32,32,36,36,39,39,43,43,
48,48,-1
1050 DATA 0,6,9,9,12,16,19,23,26,26,29
,29,32,36,39,42,48,48,-1
```

SPACE ATTACK

(16K CASSETTE, 24K DISK)

You are the commander of a stationary space station, threatened by asteroids and hostile aliens moving in on you from four directions. You must destroy them before they collide with you.

Push a joystick in port 1 in the direction you wish to fire. If your score is impressive, you can enter your name in the 'Hall of Fame'.

```
5 REM *** SPACE ATTACK
6 REM *** by G. Ryan
10 GOTO 140
19 REM MOVE ALIENS
20 COLOR 32:PLOT A1,12:PLOT A2,12:A1=A
1-D1:A2=A2+D2:COLOR C1:PLOT A1,12:COLO
R C2:PLOT A2,12:COLOR 32
30 PLOT 20,A3:PLOT 20,A4:A3=A3+D3:A4=A
4-D4:COLOR C3:PLOT 20,A3:COLOR C4:PLOT
 20,A4
40 IF PEEK(53770)<DF THEN D1=SP1:D2=SP
2:D3=SP3:D4=SP4:C1=CH:C2=C1:C3=C1:C4=C
1
50 GOTO USR(1629,KL AND A1>23 AND A2<1
6.5 AND A3<10.5 AND A4>13)
59 REM SHOOT LASERS
60 SOUND 0,35,10,8:COLOR 18:PLOT 24,12
:DRAWTO A1,12:C1=32:KL=KL-SGN(D1):D1=0
:COLOR 32:PLOT 24,12:DRAWTO A1,12
70 SOUND 0,0,0,0:A1=S1:GOTO 20
80 SOUND 0,35,10,8:COLOR 18:PLOT 16,12
```

```
:DRAWTO A2,12:C2=32:KL=KL-SGN(D2):D2=0
:COLOR 32:PLOT 16,12:DRAWTO A2,12
90 SOUND 0,0,0,0:A2=S2:GOTO 20
100 SOUND 0,35,10,8:COLOR 124:PLOT 20,
10:DRAWTO 20,A3:C3=32:KL=KL-SGN(D3):D3
=0:COLOR 32:PLOT 20,10:DRAWTO 20,A3
110 SOUND 0,0,0,0:A3=S3:GOTO 20
120 SOUND 0,35,10,8:COLOR 124:PLOT 20,
14:DRAWTO 20,A4:C4=32:KL=KL-SGN(D4):D4
=0:COLOR 32:PLOT 20,14:DRAWTO 20,A4
130 SOUND 0,0,0,0:A4=S4:GOTO 20
139 REM INITIALIZE ROUTINE
140 POKE 82,0:POKE 83,39:GOSUB 800:AD=
1570:RESTORE 1000:TRAP 160
150 FOR AD=1570 TO 1777:READ D:POKE AD
,D:NEXT AD
159 REM DOWNLOAD CHARACTER-SET
160 POP :CHSET=(PEEK(106)-12)*256:D=US
R(1570,57344,58367,CHSET):RESTORE 2000
:TRAP 190
169 REM POKE IN NEW CHARACTERS
170 READ D:FOR AD=CHSET+D TO CHSET+D+7
:READ D:POKE AD,D:NEXT AD:GOTO 170
190 DIM NAME$(120),HS(12),CH(4),BL(4),
SP(5),ST(5),BK(4)
200 FOR A=0 TO 12:HS(A)=0:NEXT A:ST(0)
=0:ST(1)=1:ST(2)=1:ST(3)=2:ST(4)=3:ST(
5)=4
210 CH(0)=0:CH(1)=64:CH(2)=96:CH(3)=64
:CH(4)=64:BL(0)=32:BL(1)=64:BL(2)=112:
BL(3)=96:BL(4)=0
220 NAME$(1)=" ":NAME$(120)=" ":NAME$(
2)=NAME$
230 SP(0)=0.5:SP(1)=0.5:SP(2)=0.6:SP(3
)=0.6:SP(4)=1.2:SP(5)=1.5:BK(0)=161:BK
(1)=34:BK(2)=163:BK(3)=154:GOTO 900
239 REM HI SCORE SCREEN
240 GRAPHICS 17:? #6;"    space attack
":? #6;"   !TOP TEN SCORES!"
250 FOR A=0 TO 9:? #6:? #6;" 00000";:P
```

```
OKE 85,PEEK(85)-LEN(STR$(HS(A))):? #6;
HS(A);"  ";NAME$(A*12+1,A*12+12):NEXT A

260 RETURN
279 REM START OF GAME
280 WV=0:SC=0:LV=3:EN=50:DF=40:SP=0:ST
=0:N=0:EX=1000:GRAPHICS 0:POKE 559,0
289 REM SET UP NEW DISPLAY LIST, IN HI
-RES. MULTICOLOR TEXT MODE
290 POKE 1536,112:POKE 1537,112:POKE 1
538,112:POKE 1539,66:POKE 1540,PEEK(88
)
300 POKE 1541,PEEK(89):FOR AD=1542 TO
1564:POKE AD,4:NEXT AD:POKE AD,65:POKE
 1566,0:POKE 1567,6
310 GRAPHICS 0:POKE 752,1:SETCOLOR 2,0
,0:POSITION 11,11:? "|PREPARE FOR ATTA
CK!|"
320 FOR A=1 TO 3:SOUND 0,100,10,8:POKE
 755,2:SETCOLOR 2,3,2:FOR D=1 TO 90:NE
XT D:N=1
330 SOUND 0,160,10,8:POKE 755,0:SETCOL
OR 2,0,0:FOR D=1 TO 90:NEXT D:NEXT A:S
OUND 0,0,0,0:POKE 755,2
339 REM INCREASE DIFFICULTY
340 WV=WV+1:POKE 559,0:POKE 1536,112:G
OSUB 960:POKE 560,0:POKE 561,6:POKE 75
6,CHSET/256
350 POKE 77,0:IF PEEK(53279)=3 THEN 85
0
360 POKE 559,34:SETCOLOR 0,8,6:SETCOLO
R 1,3,10:SETCOLOR 2,14,2:SETCOLOR 3,11
,10
370 FOR A=0 TO 18:COLOR BK(RND(0)*3):P
LOT A,RND(0)*22+1:PLOT A+20,RND(0)*22+
1:NEXT A:IF N THEN 420
380 DF=DF+10:IF DF>160 THEN DF=70
390 IF WV>2 THEN SP=SP+1:IF SP>5 THEN
SP=3
```

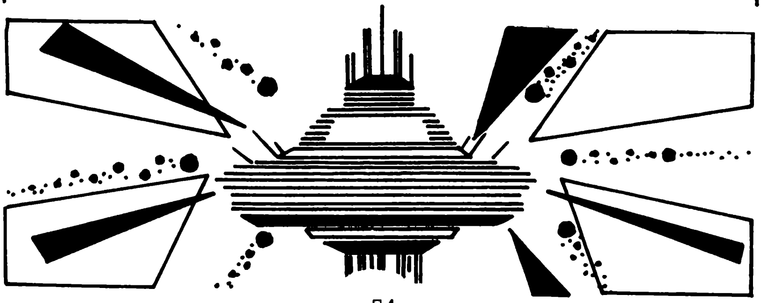
```
400 IF WV>2 THEN ST=ST+1:IF ST>5 THEN
ST=2
409 REM SET ENEMY ATTACK SPEEDS, START
ING POSITIONS, ETC...
410 EN=EN+INT(RND(0)*10+10):IF EN>200
THEN EN=90
420 S1=38-ST(RND(0)*ST):S2=1+ST(RND(0)
*ST):S3=1+ST(RND(0)*ST):S4=23-ST(RND(0
)*ST)
430 N=0:C1=32:C2=C1:C3=C1:C4=C1:A1=38:
A2=0:A3=1:A4=23:KL=EN
440 SP1=SP(RND(0)*SP):SP2=SP(RND(0)*SP
):SP3=SP(RND(0)*SP):SP4=SP(RND(0)*SP)
450 CH=CH(RND(0)*4):D1=0:D2=D1:D3=D1:D
4=D1

459 REM PLAYER'S SHIP
460 POSITION 18,11:? "!jklmn!":POSITIO
N 17,12:? "|opqrstu!":POSITION 18,13:?
 "|vwxyz|"
470 POKE 207,1:POKE 208,1:POKE 209,1:G
OTO 20
480 POSITION 18,11:IF KL=0 THEN 550
489 REM EXPLOSION
490 ? "→TPTPP←":POSITION 17,12:? "→PTP
TPTT←":POSITION 18,13:? "→PTPTT←":D=RN
D(0)*14:IF SC>EX THEN GOSUB 990
499 REM →P← IS <CTRL-P>, →T← IS <CTRL-
T>
500 FOR A=15 TO 0 STEP -0.3:SOUND 0,15
0,8,A:SOUND 1,255,2,A:SOUND 2,180,0,A:
POKE 1536,BL(RND(0)*4)

510 SETCOLOR 4,D,A+RND(0)*3:NEXT A:SC=
SC+EN-KL:LV=LV-1:POKE 712,0:IF LV<1 TH
EN 600
520 FOR A=1 TO 100:IF PEEK(53279)<>3 T
HEN NEXT A:WV=WV-1:GOTO 310
530 GOTO 850
549 REM ATTACK WAVE OVER
550 GRAPHICS 0:POKE 752,1:SETCOLOR 2,1
```

```
2,2:POSITION 8,11:? "Attack Wave ":WV;
" Completed"
560 POKE 53768,4:POKE 53761,168:POKE 5
3765,176:POKE 53760,254:POKE 53764,127
:SC=INT(SC+EN+EN*SP+INT(RND(0)*100))
570 IF SC<EX THEN FOR A=0 TO 550:NEXT
A:SOUND 0,0,0,0:SOUND 2,0,0,0:GOTO 340

580 GOSUB 980:FOR A=1 TO 550:NEXT A:SO
UND 0,0,0,0:SOUND 2,0,0,0:GOTO 340
600 FOR A=1 TO 300:NEXT A:GRAPHICS 0:S
ETCOLOR 2,0,0:POKE 752,1:GOSUB 960:POS
ITION 15,7
609 REM GAME OVER
610 ? "GAME OVER":POSITION 11,10:? "Yo
ur Score : 00000";
620 POKE 85,PEEK(85)-LEN(STR$(SC)):? S
C:POSITION 7,12
630 ? "The High Score : 00000";:A=SC:I
F A<HS(0) THEN A=HS(0)
640 POKE 85,PEEK(85)-LEN(STR$(A)):? A:
FOR X=1 TO 600:NEXT X
650 IF SC=0 THEN 900
660 FOR A=0 TO 9:IF SC<HS(A) THEN NEXT
 A:GOTO 900
670 FOR D=9 TO A+1 STEP -1:HS(D)=HS(D-
1):NAME$(D*12+1,D*12+12)=NAME$((D-1)*1
2+1,(D-1)*12+12):NEXT D:HS(A)=SC
679 REM ENTER PLAYER'S NAME
680 GRAPHICS 0:SETCOLOR 2,14,0:POKE 75
2,1:GOSUB 960:POSITION 14,6:? "GREAT S
CORE!"
690 POSITION 10,8:? "Now enter your na
me":POSITION 7,10:? "For the high scor
e board"
700 POSITION 12,12:? "↑QRRRRRRRRRRRRRR
Q←":CLOSE #5:OPEN #5,4,0,"K":GOSUB 950

709 REM ↑Q← IS <CTRL-Q>, ↑R← IS <CTRL-
R>
```

```
710 POSITION 12,13:? "|            |
":POSITION 12,14:? "→ZRRRRRRRRRRRRC←
"
719 REM →Z← IS (CTRL-Z), →C← IS (CTRL.-
C)
720 TRAP 720:Z=14:POSITION 13,13:POKE
752,0:? " ";:POKE 764,255
730 POKE 694,0:POKE 702,64:GET #5,X:IF
 X>30 AND X<96 AND Z<26 THEN POKE 767,
0:? CHR$(X);:Z=Z+1:GOTO 730
740 IF (X=126 OR X=30) AND Z>14 THEN ?
 CHR$(X);:Z=Z-1:GOTO 730
750 IF X=125 OR X=156 THEN POKE 752,1:
GOTO 710
760 IF X=127 THEN Z=25:POSITION 24,13:
? CHR$(31);:GOTO 730
770 IF X<>155 THEN 730
780 POKE 752,1:? :FOR Z=1 TO 12:LOCATE
 13+Z,13,X:NAME$(A+12+Z,A+12+Z)=CHR$(X
):NEXT Z
790 CLOSE #5:TRAP 40000:GOTO 900
800 GRAPHICS 18:POSITION 4,4:? #6;"spa
ce attack"
810 POSITION 1,6:? #6;"WRITTEN BY G. R
YAN"
820 RETURN
850 GRAPHICS 18:POSITION 2,4:? #6;"mis
sion aborted"
860 POSITION 5,6:? #6;"|game over|"
870 FOR A=1 TO 250:NEXT A
900 FOR A=53760 TO 53767:POKE A,0:NEXT
 A
910 GOSUB 240:FOR A=1 TO 500:IF STRIG(
0) AND PEEK(53279)<>6 THEN NEXT A:GOTO
 930
920 GOTO 280
930 GOSUB 800:FOR A=1 TO 150:IF STRIG(
0) AND PEEK(53279)<>6 THEN NEXT A:GOTO
 910
940 GOTO 280
```

```
950 POKE 53768,5:POKE 53761,168:POKE 5
3765,176:POKE 53760,254:POKE 53764,127
:RETURN
960 ? CHR$(125);"   SCORE 00000   WAVE
       LIVES";:POKE 85,14-LEN(STR$(SC)
):? SC
970 POSITION 22,0:? WV:POSITION 34,0:?
 LV:RETURN
980 POSITION 15,15:? "EXTRA SHIP!"
990 LV=LV+1:EX=EX+5000:RETURN
999 REM DATA ( MACHINE CODE )
1000 DATA 104,104,133,204,104,133,203,
104,133,206,104,133,205,104,133,208,10
4,133,207,160,0,177,203,145,207
1010 DATA 230,203,208,2,230,204,165,20
4,197,206,208,6,165,203,197,205,240,10
,230,207,208,230,230,208,160
1020 DATA 0,240,224,177,203,200,145,20
7,96
1030 DATA 104,104,133,213,173,197,2,17
4,199,2,141,199,2,142,197,2,104,240,43
,174,120,2,189,161,6
1040 DATA 197,209,240,16,133,209,164,2
08,132,207,133,212,173,31,208,41,252,2
40,9,96,198,207,16,242,169
1050 DATA 20,208,238,169,3,133,213,169
,82,133,212,96,230,213,169,224,208,223
,20,20,20,20,20,60,60
1060 DATA 60,20,80,80,80,20,120,100,20
,-1
1499 REM
1999 REM CHARSET DATA
2000 DATA 256,65,195,20,170,20,195,65,
0
2010 DATA 512,65,20,56,20,150,85,20,65

2020 DATA 640,25,68,17,134,36,226,68,5
0
2030 DATA 672,48,66,45,146,41,82,198,3
5
```

```
2040 DATA 768,52,141,250,39,104,158,19
3,24
2050 DATA 848,0,0,0,0,0,1,42,255
2060 DATA 856,0,0,0,5,85,85,170,255
2070 DATA 864,56,20,85,85,85,85,170,25
5
2080 DATA 872,0,0,0,80,94,85,170,255
2090 DATA 880,0,0,0,0,0,64,168,255
2100 DATA 888,1,5,85,181,181,85,5,1
2110 DATA 896,85,85,85,118,118,85,85,8
5
2120 DATA 904,85,85,85,159,159,85,85,8
5
2130 DATA 912,85,85,85,125,125,85,85,8
5
2140 DATA 920,85,85,85,167,167,85,85,8
5
2150 DATA 928,85,85,85,217,217,85,85,8
5
2160 DATA 936,64,80,85,94,94,85,80,64
2170 DATA 944,255,42,1,0,0,0,0,0
2180 DATA 952,255,170,85,85,5,0,0,0
2190 DATA 960,255,170,85,85,85,85,20,4
4
2200 DATA 968,255,170,85,85,80,0,0,0
2210 DATA 976,255,168,80,0,0,0,0,0
2220 DATA 8,0,0,0,0,0,0,0,192
2230 DATA 16,0,0,0,0,0,4,0,0
2240 DATA 24,0,0,0,64,0,0,0,0
2250 DATA 32,0,192,0,0,0,0,0,0
2260 DATA 992,12,12,12,12,12,12,12,12
```



GOBBLER

(16K CASSETTE, 24K DISK)

You are the gobbler and you have been trapped inside a dark, eerie mansion, when you sight an inhuman glow moving towards you. This glow comes from four coloured ghosts, angry at you for trespassing on their land. Your only hope is to gobble all the round pills before your pursuers catch you. When you have gobbled a power pill you become temporarily invulnerable and can eat up your pursuers to gain bonus points. You gain ten points, multiplied by the stage you've reached, for every round pill you gobble.

When you POKE in the machine language, or when the ghosts catch you three times, the program shifts to a special screen, displaying the high score. Press SELECT to change the stage you begin on, and START to begin gobbling. Use a joystick in port 1 to move up, down, left and right.

```
10 REM ********GOBBLER**********
20 REM **By Cliff McConnell
30 REM **ALL REMS MAY BE OMMITTED
40 VFLAG=1686:CDELAY=1687:CHANGE=1688:
CHANGE2=1689:SINC=1692:HITCLR=53278:CO
NSOL=53279:SPDUPFLAG=1694
50 HITEG0=1677:HPOSG0=1665:PPOS=1664:P
HITE=1676:PSPEED=1669:GSPEED=1670:DEDF
LAG=1682:DOTFLAG=1683:CHAR=1684
110 A=PEEK(740):POKE 106,A-5:START=(A-
12)*256:VBLANK=START+57:MYPMBASE=START
+1024:HS=0:DIM X(4),Y(4)
115 FOR K=1 TO 4:READ X,Y:X(K)=X:Y(K)=
Y:NEXT K
117 DATA 1,1,0,0,1,0,0,1
```

```
120 GRAPHICS 18:POSITION 7,4:? #6;"GOB
BLER":POSITION 1,6:? #6;"by cliff mcco
nnell"
130 GOSUB 30210
140 COLOR 170:PLOT 0,3:DRAWTO 19,3:DRA
WTO 19,7:DRAWTO 0,7:DRAWTO 0,3
150 GOSUB 30010
160 POKE 710,0
170 GRAPHICS 0:DL=PEEK(560)+256*PEEK(5
61):POKE DL+3,70:POKE DL+6,6:POKE DL+7
,6:POKE DL+8,6
210 POKE 704,138:POKE 705,138:POKE 706
,138:POKE 707,138:POKE 711,46:POKE 710
,0:POKE 752,1
220 POKE 559,62:POKE 1681,MYPMBASE/256
:POKE 54279,MYPMBASE/256:POKE 53277,3
310 POKE CDELAY,4:POKE CHAR,0:POKE SIN
C,2
320 LEVEL=1
510 IF SC>HS THEN HS=SC
515 POSITION 0,0:? #6;"score";SC;"
  ":POSITION 11,0:? #6;"high";HS;"
 "
520 POSITION 20,0:? #6;"PRESS START/SE
LECT"
525 STAGE=LEVEL:POSITION 0,1:? #6;"STA
GE ";STAGE:IF PEEK(CONSOL)=5 THEN POSI
TION 32,0:? #6;"select"
527 IF PEEK(CONSOL)=5 THEN 527
528 POSITION 32,0:? #6;"SELECT"
530 IF PEEK(CONSOL)=7 THEN 530
540 IF PEEK(CONSOL)=5 THEN LEVEL=LEVEL
*(LEVEL<9)+1:GOTO 525
550 IF PEEK(CONSOL)<>6 THEN 530
610 POSITION 26,0:? #6;"start"
620 IF PEEK(CONSOL)=6 THEN 620
625 SC=0:GOBS=3
626 ? CHR$(125):PILLS=4+STAGE*2:COLOR
20:POSITION 26,0:? "STAGE ";STAGE:PPIL
LS=1+INT(STAGE/10)
```

```
627 FOR K=1 TO PILLS+PPILLS:SOUND 0,25
5-(K*255/(PPILLS+PILLS)),10,10
628 X=INT(RND(0)*39):Y=INT(RND(0)*19)+
2:LOCATE X,Y,A:IF A<>32 THEN 628
629 IF K>PILLS THEN COLOR 0
635 PLOT X,Y:NEXT K:SOUND 0,0,0,0
637 POKE 704,46:POKE 705,78:POKE 706,1
42:POKE 707,206:POKE 1692,2
640 FOR K=0 TO 3:POKE HITEG0+K,50:POKE
 HPOSG0+K,50+K*50:NEXT K:POKE 1703,0
645 POSITION 0,0:? #6;"score";SC:POSIT
ION 12,0:? #6;"stage";STAGE
660 G=(4+STAGE):K=(2+STAGE*1.2):A=130-
STAGE*3:Q=INT(STAGE/4)
670 IF G>47 THEN G=47
680 IF K>47 THEN K=47
690 IF A<6 THEN A=STAGE
700 IF Q>10 THEN Q=INT(Q/2):GOTO 700
710 POKE CHANGE,A:POKE CHANGE2,Q
720 POKE PSPEED,0:POKE GSPEED,0:POKE P
HITE,200:POKE PPOS,120:POKE VFLAG,1
730 POKE 711,46:A=USR(START):? #6;"
    ":FOR Q=0 TO 200:NEXT Q:POKE PSPEE
D,G:POKE GSPEED,K
740 POKE SPDUPFLAG,6
750 POKE HITCLR,0:POKE DEDFLAG,0:POKE
DOTFLAG,0:POSITION 26,0:? "         ":
POKE 623,0
760 POKE 19,0:POKE VFLAG,0:A=USR(START
+492):POKE VFLAG,1:GOTO 1000+1000*PEEK
(1693)
1000 REM **PLAYER DIED
1010 GOBS=GOBS-1:GOSUB 1510
1020 IF GOBS=0 THEN POSITION 25,0:? "G
AME OVER":FOR G=0 TO 300:NEXT G:GOTO 5
10
1030 POSITION 22,0:? "CHOMPERS LEFT:";
GOBS:FOR G=0 TO 300:NEXT G:POSITION 22
,0:? "                "
1040 GOTO 640
```

```
1510 POKE 623,16:FOR T=15 TO 0 STEP -0
.2:POKE 711,T:SOUND 0,255-T*17,10,10:N
EXT T
1520 FOR T=250 TO 0 STEP -5:SOUND 0,T,
10,10:NEXT T:SOUND 0,0,0,0:RETURN
2000 REM **PLAYER ATE DOT
2010 POKE 623,0:X=INT((PEEK(PPOS)-46)/
4):Y=INT((PEEK(PHITE)-46)/8)
2011 FOR T=1 TO 4:SOUND 0,T*55,10,10
2013 LOCATE ABS(X+X(T)),ABS(Y+Y(T)),A:
IF A=32 THEN NEXT T:GOTO 750
2014 SOUND 0,0,0,0
2015 IF A=0 THEN 2110
2020 IF A=20 THEN COLOR 32:PLOT X+X(T)
,Y+Y(T):PILLS=PILLS-1:IF PILLS=0 THEN
STAGE=STAGE+1:SC=SC+STAGE*10:GOTO 626
2030 SC=SC+STAGE*10:POSITION 0,0:? "sc
ore";SC:GOTO 750
2110 POKE 1703,1:N=1000/STAGE:HI=INT(N
/256):POKE 1702,HI:POKE 1701,N-HI*256:
POKE 1704,1
2120 COLOR 32:PLOT X+X(T),Y+Y(T):BONUS
=200
2130 POKE 704,120:POKE 705,120:POKE 70
6,120:POKE 707,120:POKE GSPEED,PEEK(PS
PEED)-2:POKE 1692,10
2140 GOTO 750
3000 REM **PILL RAN OUT
3010 POKE 704,46:POKE 705,78:POKE 706,
142:POKE 707,206
3020 POKE 1703,0:POKE 1692,2:GOTO 750
4000 REM **ATE GHOST
4010 N=0:LOC=53257:IF PEEK(53257)=0 TH
EN LOC=53258
4020 IF PEEK(LOC)>7 THEN N=3:GOTO 4100

4030 IF PEEK(LOC)>3 THEN N=2:GOTO 4100

4040 IF PEEK(LOC)>1 THEN N=1
4100 SC=SC+BONUS
```

```
4107 POSITION 28,0:? BONUS
4110 FOR T=150 TO 0 STEP -2:SOUND 0,T,
10,10:NEXT T:POSITION 28,0:? #6;"
 ";:BONUS=BONUS+2
4120 POSITION 0,0:? #6;"score";SC:POKE
 HPOSG0+N,60:IF PEEK(PPOS)<120 THEN PO
KE HPOSG0+N,185
4125 POKE 53248+N,PEEK(HPOSG0+N):POKE
HITCLR,0
4130 GOTO 750
20000 REM *DATA FOR GHOSTS AND PLAYER
20010 DATA 0,0,0,56,124,124,124,254,25
4,254,254,170,170,0,0,0,0,0,0,56,124,1
24,84,214,254,254
20011 DATA 254,84,84,0,0,0,0,0,0,56,12
4,124,84,214,254,254,254,170,170,0,0,0
,0,0,0,56
20012 DATA 124,124,84,214,254,254,254,
84,84,0,0,0,0,0,0,24,60,126,255,255,25
5,255,126,60,24,0
20013 DATA 0,0,0,0,0,24,60,102,231,195
,195,231,102,60,24,0,0,0,0,0,0,24,60,1
02,195,129
20014 DATA 129,195,102,60,24,0,0,0,0,0
,0,24,60,102,231,195,195,231,102,60,24
,0,0,0,-1
30000 REM *POKE IN MACHINE LANGUAGE
30010 RESTORE 32000:N=0
30020 READ A:IF A<>-1 THEN POKE START+
N,A:N=N+1:GOTO 30020
30030 HI=INT(VBLANK/256):POKE START+49
,HI:POKE START+51,VBLANK-HI*256

30040 RETURN
30200 REM *POKE IN CHARACTER DATA
30210 RESTORE 20010:N=0
30220 READ A:IF A<>-1 THEN POKE 1536+N
,A:N=N+1:GOTO 30220
30230 RETURN
32000 DATA 104,160,0,173,145,6,24,105,
3,133,204,169,0,133,203,145,203,136,20
8,251,230,204,145,203,136,208
32001 DATA 251,230,204,145,203,136,208
,251,230,204,145,203,136,208,251,230,2
04,145,203,136,208,251,162,128,160,57
32002 DATA 169,6,76,92,228,173,150,6,2
40,3,76,95,228,173,0,211,41,15,141,135
,6,162,3,173,10,210
32003 DATA 45,152,6,208,93,169,15,157,
136,6,173,10,210,45,153,6,240,72,189,1
41,6,24,105,8,205,140
32004 DATA 6,144,15,56,233,16,205,140,
6,144,12,169,14,157,136,6,208,5,169,13
,157,136,6,189,129,6
32005 DATA 24,105,2,205,128,6,144,19,5
6,233,4,205,128,6,144,19,189,136,6,41,
11,157,136,6,24,144
32006 DATA 8,189,136,6,41,7,157,136,6,
24,144,8,173,10,210,41,15,157,136,6,20
2,16,152,173,145,6
32007 DATA 24,105,7,133,204,162,4,173,
134,6,224,0,208,3,173,133,6,72,41,240,
74,74,74,74,133,205
32008 DATA 104,41,15,24,125,159,6,157,
159,6,201,16,144,7,230,205,41,15,157,1
59,6,189,135,6,106,72
32009 DATA 176,15,189,140,6,56,229,205
,201,40,176,2,169,40,157,140,6,104,106
,72,176,15,189,140,6,24
32010 DATA 101,205,201,200,144,2,169,2
00,157,140,6,104,106,72,176,15,189,128
,6,56,229,205,201,45,176,2
32011 DATA 169,45,157,128,6,104,106,17
6,15,189,128,6,24,101,205,201,200,144,
2,169,200,157,128,6,202,16
32012 DATA 132,169,6,133,207,162,4,189
,140,6,133,203,173,148,6,224,0,208,3,2
4,105,64,133,206,160,15
32013 DATA 177,206,145,203,136,16,249,
198,204,202,16,225,173,128,6,141,7,208
```

```
,24,105,2,141,6,208,24,105
32014 DATA 2,141,5,208,24,105,2,141,4,
208,162,3,189,129,6,157,0,208,202,16,2
47,173,9,208,24,109
32015 DATA 10,208,240,6,238,146,6,238,
150,6,173,1,208,24,109,2,208,240,3,238
,147,6,234,234,234,206
32016 DATA 151,6,208,16,169,4,141,151,
6,173,148,6,24,105,16,41,48,141,148,6,
173,154,6,201,64,176
32017 DATA 5,160,0,140,155,6,201,112,1
44,5,160,1,140,155,6,173,155,6,208,9,1
73,154,6,24,109,156
32018 DATA 6,208,7,173,154,6,56,237,15
6,6,141,154,6,141,0,210,169,162,141,1,
210,76,95,228
32019 DATA 104,173,167,6,240,107,165,2
0,205,164,6,240,100,141,164,6,206,165,
6,208,8,173,166,6,240,3
32020 DATA 206,166,6,173,166,6,208,79,
173,165,6,201,160,176,72,201,0,208,9,1
69,2,141,157,6,141,150
32021 DATA 6,96,206,168,6,208,54,169,1
0,141,168,6,173,192,2,201,72,240,19,16
9,72,141,192,2,141,193
32022 DATA 2,141,194,2,141,195,2,208,1
7,24,144,167,169,120,141,192,2,141,193
,2,141,194,2,141,195,2
32023 DATA 173,133,6,56,233,2,141,134,
6,173,146,6,240,13,169,3,174,167,6,208
,2,169,0,141,157,6
32024 DATA 96,173,147,6,240,9,169,1,14
1,157,6,141,150,6,96,165,19,205,158,6,
208,191,169,0,133,19
32025 DATA 238,134,6,24,144,181,-1
```

GRAND PRIX

(16K CASSETTE, 24K DISK)

You are an expert car driver, trying to steer through the most treacherous race course you have ever seen. This course is in five stages and at the end of each stage the track gets narrower and winds more dramatically.

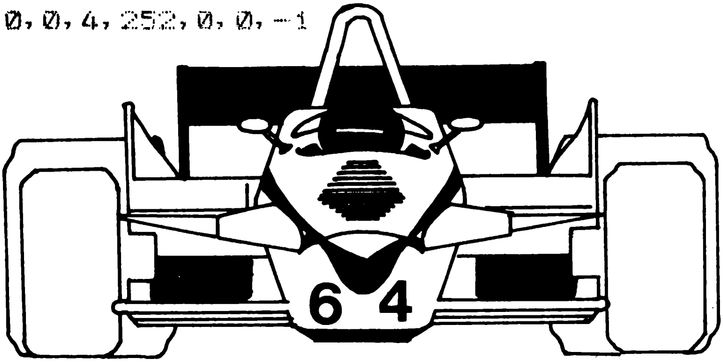
You start the game with five cars. When you have crashed all five cars, your skill rating will be displayed (the maximum rating is 100). If you succeed in completing the course you are put onto an even harder one. Use a joystick in port 1 to move left and right.

```
0 REM *** GRAND PRIX, BY G. RYAN
10 GOTO 400
20 Z=USR(1540):? TL$;:POKE 85,PEEK(204
)+W:? TR$;:IF  NOT Z THEN 20
30 IF Z>255 THEN 200
39 REM NEXT STAGE
40 ST=ST+1:SI=SI+2:W=WD(ST):POKE 1539,
RAND(ST):POKE 77,0:SOUND 1,155-ST*3,2,
6:IF ST=6 THEN 110
50 TL$=L$(SI):TR$=R$(SI):TR$(3)=CHR$(1
57):IF ST>4 THEN 110
60 IF PEEK(204)>10 THEN POSITION 3,0
70 IF PEEK(204)<11 THEN POSITION 30,0
80 POKE 1536,LN(ST):POKE 205,0:POKE 53
278,0
90 ? "Stage ";ST+1;
100 GOTO 20
110 IF ST=6 THEN 130
119 REM FINISHED
120 POKE 1536,17:POKE 205,0:SETCOLOR 4
,10,0:POSITION PEEK(204)+2,0:? "FINISH
":GOTO 20
```

```
130 SOUND 0,0,0,0:SOUND 1,0,0,0:FOR A=
1 TO 200:NEXT A
140 CS=CS+1:IF BN>HS THEN HS=BN
150 ? CHR$(125):POSITION 11,8:? "Cours
e ";CS;" Completed":POSITION 15,10:? "
Crashes ";CR:SOUND 1,255,2,4
160 POKE 53248,120:POSITION 12,13:? "S
kill Rating ";BN:POSITION 8,15:? "High
est Skill Rating ";HS
170 POSITION 2,17:? "Hit START when re
ady for next course"
180 IF PEEK(53279)<>6 AND STRIG(0) THE
N 180
190 WD(0)=10:WD(1)=9:RAND(4)=45:IF ST>
2 THEN WD(2)=8
195 GOTO 680
199 REM CRASHED
200 SOUND 1,0,0,0:FOR A=1 TO 15:NEXT A
:PM$(IN+YP)=E$
210 FOR A=15 TO 0 STEP -0.5:SOUND 0,RN
D(0)*50+100,8,A:POKE 704,48+A:NEXT A
220 CR=CR+1:BN=BN-(6-ST)*(CR+1):BN=BN*
(BN>0)
230 PM$(2)=PM$:POKE 704,CC:IF CR<5 THE
N SI=SI-2:ST=ST-1:GOTO 690
239 REM GAME OVER
240 POSITION 15,10:? "GAME OVER":POSIT
ION 13,4:BN=INT(ST*4+BN/50+0.5)
250 IF BN>HS THEN HS=BN
260 ? "Skill Rating ";BN:POSITION 9,7:
? "Highest Skill Rating ";HS:POSITION
15,13:? "Hit START":CS=0
270 IF PEEK(53279)<>6 AND STRIG(0) THE
N 270
280 GOTO 680
400 GRAPHICS 18:POSITION 5,4:? #6;"gra
nd prix"
410 POSITION 1,6:? #6;"WRITTEN BY G. R
YAN"
```

```
420 FOR A=53248 TO 53265:POKE A,0:NEXT
 A
499 REM INITIALIZING
500 DIM PM$(1200),C$(24),E$(24),TL$(2)
,TR$(3),L$(13),R$(13)
510 PM$(1)=CHR$(0):PM$(1200)=CHR$(0):P
M$(2)=PM$
520 RESTORE 5000:TRAP 540:FOR A=1540 T
O 1777:READ D:POKE A,D:NEXT A
540 DIM WD(6),RAND(6),LN(6)
560 A=ADR(PM$):D=INT(A/1024)*1024+512:
IF D<A THEN D=D+1024
570 IN=D-A:POKE 559,46:POKE 53277,3:PO
KE 54279,(D-512)/256
580 RESTORE 590:FOR A=0 TO 5:READ B,C,
D:WD(A)=B:RAND(A)=C:LN(A)=D:NEXT A
587 REM DATA FOR TRACK WIDTH, DIRECTIO
N CHANGES, LENGTH OF STAGE
590 DATA 11,45,210,10,50,230,9,55,220,
9,55,220,8,30,120,8,20,20
600 RESTORE 620:FOR A=1 TO 12:READ D:L
$(A)=CHR$(D):NEXT A
610 FOR A=1 TO 12:READ D:R$(A)=CHR$(D)
:NEXT A
620 DATA 19,19,20,20,9,12,34,34,46,46,
19,19
630 DATA 19,19,20,20,11,15,34,34,46,46
,19,19
640 RESTORE 650:FOR A=1 TO 24:READ D:C
$(A)=CHR$(D):NEXT A
650 DATA 0,0,0,0,0,0,24,90,126,90,24,6
0,60,60,219,255,219,0,0,0,0,0,0,0
660 RESTORE 670:FOR A=1 TO 24:READ D:E
$(A)=CHR$(D):NEXT A
665 POSITION 4,10:? #6;"PRESS START...
"
667 IF PEEK(53279)<>6 AND STRIG(0) THE
N 667
670 DATA 0,0,0,0,0,16,66,8,162,9,84,36
,82,9,130,32,5,80,0,0,0,0,0,0,0
```

```
675 POKE 82,0:POKE 83,39:GRAPHICS 0:PO
KE 710,0:POKE 709,14:POKE 752,1:POKE 5
59,46:POKE 53277,3:? "  "
680 ST=-1:BN=100:SI=-1:CR=0:CC=INT(RND
(0)*15)*16+10:POKE 1536,2:POKE 1537,2:
POKE 1538,24:POKE 1539,20:YP=85
690 POKE 204,14:POKE 205,0:POKE 203,12
0:POKE 1536,1:POKE 712,0:Z=ST:IF Z<0 T
HEN Z=0
700 B=Z*2+1:? CHR$(125):FOR A=0 TO 23:
POSITION 14,A:? L$(B,B+1);:POSITION 14
+WD(Z),A:? R$(B,B+1);:NEXT A
710 PM$(IN+YP)=C$:POKE 704,CC:POKE 532
48,120:POKE 53278,0
740 GOTO 40
4999 REM DATA FOR USR SUBROUTINE
5000 DATA 104,173,120,2,74,74,41,3,170
,189,115,6,141,0,210,189,119,6,141,1,2
10,165,203,24,125,123,6
5010 DATA 133,203,141,0,208,160,0,132,
212,132,84,173,4,208,133,213,206,0,6,2
08,2,198,212,165,204,166,205
5020 DATA 24,125,111,6,133,204,133,85,
205,1,6,240,9,205,2,6,208,10,169,1,208
,2,169,2,133,205,208
5030 DATA 16,173,10,210,205,3,6,176,8,
41,7,170,189,103,6,133,205,96,0,1,2,0,
1,2,1,2,0
5040 DATA 255,1,0,0,27,27,0,0,132,132,
0,0,4,252,0,0,-1
```



ESCAPE

(16K CASSETTE, 24K DISK)

While flying your shuttle back to earth you encounter an almost insurmountable problem. Mines have been laid just about everywhere, and a yellow wall obstructs your path. You must dodge all of the mines and squeeze through the gap in the wall if you are to have any hope of seeing your family again.

Press the fire button to start your craft moving across the screen. Use a joystick in port 1 to move it up and down. If you succeed in escaping, more mines will be drawn up and you must press the fire button again. Pods situated among the mines gain you bonus points.

```
0 REM ESCAPE, written by G. Ryan
10 GOTO 500
20 S=STICK(0):YP=YP+(S=13)-(S=14):XP=X
P+1:POKE 53248,XP:PM$(YP)=PL$:IF  NOT
PEEK(53252) THEN 20
30 IF PEEK(53252)<>4 THEN 200
39 REM PLAYER HIT SOMETHING
40 X=INT((XP-48)/8):Y=INT((YP-IN-13)/4
):IF XP>176 THEN 650
50 FOR A=N TO N1:FOR B=-N1 TO N2:LOCAT
E X+A,Y+B,C:IF C<>164 THEN NEXT B:NEXT
 A:POKE 53278,N:GOTO 20
60 POP :POP :SC=SC+ST*INT(RND(N)*N2+N1
):POSITION 7,23:? #N6;SC
70 COLOR 32:PLOT X+A,Y+B:POKE 53278,N
80 FOR A=N1 TO 15:SOUND N,PEEK(53770),
10,10:NEXT A:SOUND N,N,N,N
90 POKE 53278,N:GOTO 20
100 FOR A=N TO 511:POKE D+A,PEEK(57344
+A):NEXT A:RETURN
```

```
200 FOR A=15 TO N STEP -N3:FOR D=N TO
N2
210 SOUND D,RND(N)*50+100,8,A:NEXT D:P
OKE 704,32+A:NEXT A:POKE 704,N
220 POSITION 7,23:? #N6;SC:LV=LV-N1:IF
 LV>N THEN 290
230 POSITION N3,10:? #N6;"  GAME OVER
 ":IF SC>HS THEN HS=SC
250 POSITION N2,13:? #N6;"  |hi score|
 ";HS:POSITION N3,17:? #N6;"  HIT star
t."
260 IF PEEK(53279)<>N6 AND STRIG(N) TH
EN 260
270 POKE 77,N:GOTO 580
290 XP=38:YP=IN+61:POKE 53248,XP:POKE
704,40:POKE 53278,N:PM$(N2)=PM$
300 IF STRIG(N) THEN 300
310 GOTO 20
499 REM INITIALIZING
500 READ N,N1,N2,N3,N6,E,W:DATA 0,1,2,
3,6,255,256
510 A=PEEK(740)-4:POKE 106,A-2:GRAPHIC
S 18:FOR X=53248 TO 53255:POKE X,N:NEX
T X
520 POSITION 7,4:? #N6;"escape":POSITI
ON N1,N6:? #N6;"WRITTEN BY G. RYAN"
530 D=A*W:GOSUB 100:RESTORE 5000
540 READ A:IF A<>-N1 THEN FOR X=A+D TO
 A+D+7:READ A:POKE X,A:NEXT X:GOTO 540

550 DIM PM$(1200),PL$(15):PM$(N1)=CHR$
(N):PM$(1200)=PM$(N1):PM$(N2)=PM$
560 RESTORE 5500:FOR A=N1 TO 13:READ X
:PL$(A)=CHR$(X):NEXT A
570 A=ADR(PM$):PM=INT(A/1024)*1024:X=P
M+512:IN=X-A:IF A>X THEN PM=PM+1024:X=
X+1024:IN=IN+1024
580 YP=IN+60:GRAPHICS 17:POKE 54279,PM
/W:POKE 53277,N3:POKE 559,46
590 POKE 704,40:POKE 623,N1:POKE 756,P
```

```
EEK(106)+N2
600 SETCOLOR N,8,N6:SETCOLOR N1,N3,8:S
ETCOLOR N2,11,8
610 SETCOLOR N3,14,10
620 COLOR 35:PLOT N,N:DRAWTO 19,N:PLOT
 N,22:DRAWTO 19,22
630 COLOR 129:PLOT 18,N1:DRAWTO 18,21:
COLOR 165:PLOT 19,N1:DRAWTO 19,21
640 POSITION N1,23:? #N6;"score 0    st
age 1":LV=3:ST=N:SC=N:G=N1:M=N
650 SC=SC+M:YP=IN+61:ST=ST+1:POSITION
17,23:? #N6;ST:POSITION 7,23:? #N6;SC:
PM$(N2)=PM$
660 M=INT(RND(N)*N3)+N1:FOR A=N1 TO M:
COLOR N2:PLOT RND(N)*12+N2,RND(N)*20+N
1:COLOR 32
670 PLOT RND(N)*14+N2,RND(N)*20+N1:PLO
T RND(N)*14+N2,RND(N)*20+N1:NEXT A:COL
OR 164
680 PLOT RND(N)*12+N3,RND(N)*20+N1:COL
OR 129:PLOT 18,G:DRAWTO 18,G+N2:G=INT(
RND(N)*17+N1):COLOR 32
690 PLOT 18,G:DRAWTO 18,G+N2:YP=IN+60:
XP=38:POKE 53278,N
700 IF STRIG(N) THEN 700
710 POKE 53278,N:GOTO 20
4999 REM DATA FOR NEW CHARACTERS AND P
LAYERS SHIP
5000 DATA 8,238,238,238,14,238,238,238
,224
5010 DATA 24,0,247,247,247,0,127,127,1
27
5020 DATA 16,129,90,36,90,90,36,90,129

5030 DATA 32,0,16,56,84,254,84,56,16
5040 DATA 40,0,64,0,2,0,16,0,0
5050 DATA -1
5500 DATA 0,0,0,112,224,88,239,88,224,
112,0,0,0
```

TIME TYPIST

(16K CASSETTE, 24K DISK)

In this game, the computer prints up a random sequence of 20 letters. You must type in this sequence as quickly as possible; the timer meanwhile is ticking down to zero. Press BACK SPACE if you make a mistake. At the end of the stage, another sequence of letters is printed up and you are given less time to repeat it. This is an excellent game for anyone who wants to practise their typing and see how they are improving.

```
10 REM **TIME TYPIST
20 REM **By Cliff McConnell
110 DIM L$(30),P$(30):OPEN #1,4,0,"K"
120 GRAPHICS 18:SC=0:HS=SC
150 REM
210 POSITION 1,0:? #6;"SCORE:";SC;"
"
220 POSITION 1,1:? #6;"HIGH SCORE:";HS

230 POSITION 1,5:? #6;"please press st
art"
240 IF PEEK(53279)<>6 THEN 240
245 POSITION 1,5:? #6;"
    "
250 ST=1
270 T=15+10*(ST=1)-(ST>2)*(ST-2)
280 N=20
310 X=INT((20-N)/2):P=0
510 GOSUB 10010:POSITION 1,2:? #6;"TIM
E:";T
710 POKE 20,0:POKE 19,0:POKE 20,0
720 IF PEEK(20)>50 THEN POKE 20,0:T=T-
1
```

```
730 POSITION 6,2:? #6;T;" ";:IF  NOT T
 THEN 8010
750 IF PEEK(764)=255 THEN 720
760 GET #1,A:POKE 764,255:IF A<65 OR A
>90 THEN 1010
770 IF P<N THEN POSITION X+P,7:? #6;CH
R$(A+128):P=P+1:P$(P,P)=CHR$(A):IF P$=
L$ THEN 5010
780 GOTO 720
1010 IF A=126 AND P>0 THEN P=P-1:POSIT
ION X+P,7:? #6;" ":P$(P+1,P+1)=" ":GOT
O 720
1020 IF A=32 AND P<N THEN P=P+1:GOTO 7
20
2010 GOTO 720
5000 REM **DID IT!!
5010 SC=SC+ST*T:POSITION 7,0:? #6;SC
5090 ST=ST+1:GOTO 270
8000 REM **DIDN'T MAKE IT
8010 POSITION 5,4:? #6;"bad luck":FOR
K=0 TO 200:NEXT K:POSITION 5,4:? #6;"
       "
8020 IF SC>HS THEN HS=SC
8050 GOTO 150
10000 REM **PRINT UP LETTERS
10010 POSITION 5,4:? #6;"get ready"
10015 POSITION 0,7:? #6;"
       ":POSITION 0,6:? #6;"
          "
10018 POSITION INT((20-N)/2),6
10020 P$="":L$="":FOR K=1 TO N:A=INT(R
ND(0)*26)+65:? #6;CHR$(A+160);:L$(K,K)
=CHR$(A)
10030 SOUND 0,100,10,10:FOR G=0 TO 15:
NEXT G:SOUND 0,0,0,0:NEXT K
10050 POSITION 5,4:? #6;"          "
10110 RETURN
```

GALAX ATTACK

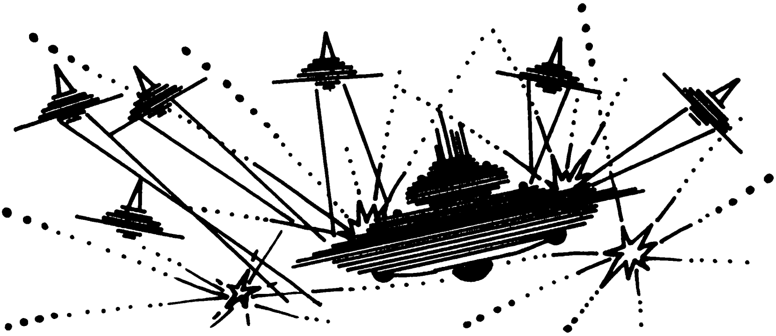
(24K CASSETTE, 32K DISK)

In this game you must destroy wave after wave of attacking sinistoids. These disagreeable aliens peel out of their formation and swoop down on your ship at the bottom of the screen, dropping deadly bombs as they go.

Destroying a mutant (a multi-coloured sinistoid which has mutated after leaving the formation) will gain you bonus points depending on the colour of the alien you hit before it. Destroying other attacking sinistoids will give you between 10 and 40 points.

Use a joystick in port 1 to move your ship left or right. Press the fire button to release a missile. After a delay, while the program POKEs in the machine language, the title screen will be printed up.

Press OPTION to change the number of bombs the sinistoids can have on the screen at once. Press SELECT to alter the level you start on, and START to begin the game. After you have been destroyed three times, the program will again return to the title screen and display the high score.



```
10 REM **GALAX ATTACK
20 REM **By Cliff McConnell
30 REM **ALL REMS MAY BE OMITTED
90 FOR T=1536 TO 1536+255:POKE T,0:NEX
T T
110 TOP=PEEK(740):POKE 106,TOP-5:POKE
0,TOP-8:START=(TOP-16)*256:CH=(TOP-18)
*256:POKE 1769,CH/256
120 DVBLANK=START:IVBLANK=START+1223:D
LI=START+1790:USER=START+1802:CLEAR=ST
ART+1822
125 GRAPHICS 18:DL=PEEK(560)+256*PEEK(
561):SCRN=PEEK(88)+256*PEEK(89):FOR G=
1 TO 11
127 TRAP 129:? #6;"*******************
*";:NEXT G
128 ? #6;"****":GOTO 130
129 ? #6:GOTO 127
130 POSITION 9,4:? #6;"galax":POSITION
 11,6:? #6;"attack"
135 POKE DL+9,PEEK(DL+11)+16:POKE 5427
6,4
140 N0=0:N1=1:N2=2:N6=6:GOTO 5010
200 REM **DATA FOR NEW CHARACTER SET
210 DATA 0,0,0,0,0,0,0,0,0,126,219,255
,129,195,126,0,8,28,62,62,127,127,127,
255,4,14
211 DATA 31,31,63,63,63,127,2,7,15,15,
31,31,31,63,1,3,7,7,15,15,15,31,0,1,3,
3
212 DATA 7,7,7,15,0,0,1,1,3,3,3,7,0,0,
0,0,1,1,1,3,0,0,0,0,0,0
213 DATA 0,1,0,0,0,0,0,0,0,128,0,0,0,0
,128,128,128,192,0,0,128,128,192,192,1
92,224
214 DATA 0,128,192,192,224,224,224,240
,128,192,224,224,240,240,240,248,64,22
4,240,240,248,248,248,252,32,112
215 DATA 248,248,252,252,252,254,16,56
,124,124,254,254,254,255,0,0,0,0,8,8,8
```

```
,8,0,0,0,0
216 DATA 4,4,4,4,0,0,0,0,2,2,2,2,0,0,0
,0,1,1,1,1,0,0,0,0,128,128
217 DATA 128,128,0,0,0,0,64,64,64,64,0
,0,0,0,32,32,32,32,0,0,0,0,16,16,16,16

218 DATA 8,8,8,8,0,0,0,0,4,4,4,4,0,0,0
,0,2,2,2,2,0,0,0,0,1,1
219 DATA 1,1,0,0,0,0,128,128,128,128,0
,0,0,0,64,64,64,64,0,0,0,0,32,32,32,32

220 DATA 0,0,0,0,16,16,16,16,0,0,0,0,1
53,90,60,255,255,60,90,153,0,127,65,65
,65,65
221 DATA 65,127,0,0,62,34,34,34,62,0,0
,0,0,28,28,28,0,0,147,153,205,103,51,2
7,15,3,201,153,179,230,204,216
222 DATA 240,192,-1
2200 REM **MACHINE LANGUAGE DATA
2210 DATA 173,156,6,240,3,76,98,228,17
3,0,211,106,106,106,176,35,173,133,6,2
01,221,144,66,206,132,6
2211 DATA 16,61,169,7,141,132,6,169,0,
172,133,6,200,145,88,172,131,6,145,88,
206,133,6,208,38,106
2212 DATA 176,35,173,133,6,201,239,176
,28,238,132,6,173,132,6,41,7,141,132,6
,208,15,169,0,172,133
2213 DATA 6,145,88,172,131,6,145,88,23
8,133,6,173,142,6,208,58,173,133,6,56,
233,20,174,132,6,168
2214 DATA 169,0,224,4,144,9,145,88,200
,208,8,240,2,240,226,200,145,88,136,14
0,131,6,173,132,6,141
2215 DATA 130,6,173,16,208,205,242,6,2
40,105,141,242,6,201,1,240,98,169,1,14
1,142,6,173,130,6,73
2216 DATA 8,141,130,6,41,8,208,81,173,
131,6,56,233,20,141,131,6,168,177,88,4
1,63,201,1,208,43
```

```
2217 DATA 177,88,24,105,64,201,64,176,
26,162,0,189,193,6,240,3,232,208,248,1
52,157,193,6,169,200,141
2218 DATA 244,6,169,4,157,203,6,169,0,
145,88,173,131,6,24,144,7,173,131,6,20
1,21,176,13,24,105
2219 DATA 20,168,169,0,145,88,141,142,
6,240,128,173,231,6,240,11,206,231,6,2
08,17,173,230,6,141,229
2220 DATA 6,206,229,6,208,6,173,232,6,
141,231,6,160,4,185,115,6,208,54,173,2
31,6,208,84,173,10
2221 DATA 210,201,150,176,77,133,203,1
52,72,164,203,177,88,133,204,41,63,201
,1,240,5,104,168,24,144,56
2222 DATA 169,0,145,88,104,168,165,203
,141,164,6,162,0,134,205,173,164,6,56,
176,2,208,102,233,20,201
2223 DATA 20,144,15,141,164,6,165,205,
24,105,20,133,205,232,224,6,144,227,18
9,16,6,153,115,6,173,164
2224 DATA 6,56,176,3,56,176,115,233,20
,74,170,189,24,6,153,120,6,238,157,6,1
73,10,210,41,15,153
2225 DATA 125,6,165,203,24,144,3,24,14
4,130,153,145,6,165,204,153,165,6,41,1
92,74,74,74,74,74,74
2226 DATA 170,189,196,2,192,0,240,3,15
3,191,2,169,100,141,243,6,169,166,141,
1,210,173,10,210,208,21
2227 DATA 173,10,210,41,1,208,8,185,12
5,6,41,254,24,144,3,25,125,6,153,125,6
,185,125,6,74,170
2228 DATA 189,134,6,41,248,74,74,74,13
3,203,189,134,6,41,7,24,144,3,24,144,3
3,121,150,6,153,150
2229 DATA 6,201,8,144,13,24,144,3,24,1
44,148,41,7,153,150,6,230,203,185,125,
6,41,1,208,24,24
2230 DATA 144,3,24,144,63,185,120,6,24
```

```
,101,203,201,200,144,29,185,125,6,9,1,
153,125,6,185,120,6
2231 DATA 56,229,203,201,45,176,11,185
,125,6,41,254,153,125,6,24,144,217,153
,120,6,173,144,6,41,248
2232 DATA 74,74,74,133,203,173,144,6,4
1,7,24,144,6,24,144,170,24,144,95,121,
159,6,153,159,6,201
2233 DATA 8,144,7,41,7,153,159,6,230,2
03,185,115,6,24,101,203,201,230,144,65
,152,72,170,188,115,6
2234 DATA 162,15,24,105,3,24,101,0,133
,204,169,0,133,203,145,203,136,202,16,
250,104,168,72,185,165,6
2235 DATA 170,185,145,6,168,138,145,88
,104,168,169,0,153,120,6,153,115,6,206
,157,6,192,0,208,5,169
2236 DATA 4,141,114,6,24,144,3,153,115
,6,136,16,152,206,191,6,208,72,169,2,1
41,191,6,162,9,189
2237 DATA 180,6,240,30,189,170,6,73,8,
157,170,6,240,20,189,180,6,24,105,20,1
57,180,6,208,9,222
2238 DATA 180,6,160,236,169,0,145,88,2
02,16,218,173,227,6,240,11,206,227,6,2
08,17,173,226,6,141,225
2239 DATA 6,206,225,6,208,6,173,228,6,
141,227,6,173,10,210,41,3,168,185,115,
6,240,56,132,203,173
2240 DATA 190,6,201,9,176,43,173,227,6
,208,67,173,10,210,41,3,208,60,162,0,1
89,180,6,240,5,232
2241 DATA 224,9,144,246,173,10,210,41,
3,208,43,173,10,210,41,255,168,192,20,
144,87,192,240,176,83,144
2242 DATA 2,240,79,177,88,41,63,201,1,
208,71,152,24,105,20,157,180,6,238,190
,6,169,8,157,170,6
2243 DATA 208,54,164,203,185,115,6,56,
233,48,74,74,74,74,168,185,34,6,133,20
```

```
4, 164, 203, 185, 120, 6, 56
2244 DATA 233,44,74,74,74,24,101,204,2
4,105,20,201,190,176,15,201,40,144,11,
157,180,6,169,8,157,170
2245 DATA 6,238,190,6,162,4,189,115,6,
240,39,189,214,6,208,34,173,142,6,73,1
,208,52,189,235,6
2246 DATA 240,78,189,115,6,56,233,52,7
4,74,74,74,168,185,34,6,133,203,189,12
0,6,56,176,3,24,144
2247 DATA 71,233,48,74,74,74,24,101,20
3,168,204,131,6,208,4,240,8,16,191,200
,204,131,6,208,64,189
2248 DATA 214,6,208,59,169,200,141,244
,6,169,1,157,214,6,169,4,157,219,6,172
,131,6,173,133,6,56
2249 DATA 176,2,240,122,233,20,141,131
,6,169,0,145,88,141,142,6,152,24,144,4
,144,119,16,197,105,20
2250 DATA 168,169,0,145,88,224,0,208,5
5,240,2,208,89,172,114,6,185,109,6,133
,203,152,10,10,10,10
2251 DATA 10,10,24,105,16,160,12,145,8
8,136,145,88,136,24,144,2,16,208,101,2
03,145,88,169,60,141,192
2252 DATA 6,169,4,141,114,6,165,203,16
0,17,208,17,189,165,6,74,74,74,74,74,7
4,141,114,6,24,105
2253 DATA 1,160,18,24,113,88,201,26,14
4,10,56,233,10,145,88,136,169,1,208,23
9,145,88,240,53,173,133
2254 DATA 6,56,233,220,133,203,189,115
,6,201,220,144,38,189,120,6,56,233,44,
56,237,132,6,74,74,74
2255 DATA 197,203,208,21,234,234,234,2
34,234,234,234,169,1,141,155,6,141,156
,6,141,224,6,76,98,228,202
 2256 DATA 16,140,173,192,6,240,17,206,
 192,6,208,12,169,0,160,10,145,88,200,1
 45,88,200,145,88,76,98
```

```
2257 DATA 228,173,155,6,240,3,76,95,22
8,162,4,189,3,208,157,235,6,202,208,24
7,162,3,169,0,24,125
2258 DATA 0,208,202,16,249,141,235,6,1
72,133,6,177,88,201,18,144,31,201,26,1
76,27,56,233,18,56,233
2259 DATA 4,41,7,205,132,6,240,2,144,1
2,238,224,6,238,155,6,238,156,6,76,95,
228,200,177,88,201
2260 DATA 18,144,31,201,26,176,27,56,2
33,18,56,233,4,41,7,205,132,6,240,2,17
6,12,238,224,6,238
2261 DATA 155,6,238,156,6,76,95,228,20
6,240,6,208,8,169,3,141,240,6,141,30,2
08,165,0,24,105,7
2262 DATA 133,204,169,0,133,203,162,4,
188,115,6,240,20,189,214,6,208,15,138,
72,162,15,189,0,6,145
2263 DATA 203,136,202,16,247,104,170,1
98,204,202,16,226,162,4,189,214,6,240,
106,138,72,189,219,6,10,10
2264 DATA 10,10,170,104,72,24,101,0,24
,105,3,133,204,169,0,133,203,104,72,17
0,188,115,6,189,219,6
2265 DATA 208,36,162,25,169,0,145,203,
136,202,16,250,104,170,206,234,6,208,9
,238,155,6,238,156,6,76
2266 DATA 95,228,169,0,157,115,6,157,2
14,6,240,35,169,15,133,205,138,24,105,
15,170,189,45,6,145,203
2267 DATA 202,136,198,205,16,245,104,1
70,206,245,6,208,8,169,2,141,245,6,222
,219,6,202,16,142,172,133
2268 DATA 6,173,132,6,24,105,2,145,88,
24,105,8,200,145,88,173,142,6,240,11,1
73,131,6,24,105,20
2269 DATA 168,169,0,145,88,172,131,6,1
73,130,6,24,105,18,24,105,64,145,88,16
2,4,189,120,6,157,255
2270 DATA 207,202,208,247,173,120,6,14
```

```
1,7,208,24,105,2,141,6,208,24,105,2,14
1,5,208,24,105,2,141
2271 DATA 4,208,162,9,188,180,6,240,93
,192,240,176,20,192,20,144,16,177,88,2
40,6,41,63,201,1,240
2272 DATA 6,189,170,6,24,144,44,188,18
0,6,177,88,41,63,201,1,240,4,169,0,145
,88,189,180,6,56
2273 DATA 233,20,168,177,88,41,63,201,
1,240,4,169,0,145,88,206,190,6,169,0,1
57,180,6,240,25,188
2274 DATA 180,6,24,105,18,145,88,152,5
6,233,20,168,177,88,41,63,201,1,240,4,
169,0,145,88,202,16
2275 DATA 155,206,213,6,208,91,169,2,1
41,213,6,162,9,188,193,6,240,34,189,20
3,6,208,21,169,0,157
2276 DATA 193,6,206,234,6,208,9,238,15
5,6,238,156,6,76,95,228,169,223,24,105
,33,145,88,222,203,6
2277 DATA 202,16,214,173,243,6,240,13,
24,105,5,144,2,169,0,141,243,6,141,0,2
10,173,244,6,240,19
2278 DATA 162,136,56,233,10,80,3,169,0
,170,141,244,6,141,2,210,142,3,210,76,
95,228,72,173,233,6
2279 DATA 141,10,212,141,9,212,104,64,
104,169,0,141,155,6,141,156,6,173,224,
6,208,5,173,234,6,208
2280 DATA 246,96,104,165,0,24,105,3,13
3,205,169,0,133,204,162,5,160,0,152,14
5,204,136,208,251,230,205
2281 DATA 202,208,246,96,0,-1
3200 REM **PAGE SIX DATA
3210 DATA 0,0,0,24,60,126,219,219,255,
195,102,60,24,0,0,0,56,72,98,104,102,1
18,134,150,58,74
3211 DATA 90,106,122,138,154,170,186,2
02,0,20,40,60,80,100,120,140,160,180,2
00,0,0,0,42,42,28,28
```

```
3212 DATA 127,127,28,28,42,42,0,0,0,0,
0,0,0,62,62,34,34,34,34,62,62,0,0,0,0,
0
3213 DATA 0,0,0,0,28,28,20,20,28,28,0,
0,0,0,0,0,0,0,0,0,0,28,28,28,28,0
3214 DATA 0,0,0,0,0,1,2,4,8,0,0,-1
5000 REM **POKE IN CHARACTERS
5010 POSITION 8,7:? #N6;"PLEASE**WAIT"
:TRAP 5110:FOR N=CH TO CH+6666:READ A:
POKE N,A:NEXT N
5100 REM **POKE IN ML
5110 POSITION 7,7:? #N6;"NOT*LONG*TO*G
O":TRAP 5210:FOR N=START TO START+6666
:READ A:POKE N,A:NEXT N
5200 REM **POKE IN PAGE 6 DATA
5210 POSITION 6,7:? #N6;"NEARLY**FINIS
HED":TRAP 5310:FOR N=1536 TO 9999:READ
 A:POKE N,A:NEXT N
5300 REM
5310 POKE 1749,N1:POKE 1727,N1:POKE 17
63,N1:POKE 1767,N1:POKE 1691,N1:POKE 1
692,N1:SC=N0:HS=N0:LEVEL=N1:BO=9
5410 GRAPHICS 18:POKE 53277,3:POKE 559
,62:POKE 54279,TOP-8:POKE 1781,2
5411 POSITION N0,N0:? #N6;"high";HS;"
         ";:POSITION 10,N0:? #N6;"scor
e";SC;"      "
5412 POSITION N0,N1:? #N6;"press:":? #
N6;"start TO START GAME"
5414 ? #NE;"select LEVEL:";LEVEL
5416 ? #NE;"option BOMBS:";BO
5420 IF PEEK(53279)=6 THEN 10001
5430 IF PEEK(53279)=5 THEN 6010
5440 IF PEEK(53279)<>3 THEN 5420
5450 REM **PRESSED OPTION
5460 BO=(BO+N1)*(BO<9):POSITION 13,4:?
 #NE;BO
5470 IF PEEK(53279)=3 THEN 5470
5480 GOTO 5420
6000 REM **PRESSED SELECT
```

```
6010 LEVEL=LEVEL*(LEVEL<9)+1:POSITION
13,3:? #N6;LEVEL;" "
6020 IF PEEK(53279)=5 THEN 6020
6030 GOTO 5420
10000 REM **START GAME
10001 POKE 0,TOP-8:A=USR(CLEAR)
10005 ? #N6;CHR$(125):POSITION 15,N0:?
 #N6;"00000"
10007 POKE 54286,N0:HI=INT(IVBLANK/256
):POKE 547,HI:POKE 546,IVBLANK-HI*256:
HI=INT(DVBLANK/256):POKE 549,HI
10008 POKE 548,DVBLANK-HI*256:HI=INT(D
LI/256):POKE 513,HI:POKE 512,DLI-HI*25
6:POKE 54286,192
10009 POKE DL+3,PEEK(DL+3)+128
10010 SC=N0:SHIPS=N2+N1:STAGE=LEVEL
11010 ALIENO=20+STAGE*N2:K=8:N=0.8+STA
GE*0.2:IF N>3 THEN N=3
11020 IF K>16 THEN K=16
11030 IF ALIENO>40 THEN ALIENO=40
11040 POKE 1680,K:POKE 1770,ALIENO:FOR
 K=0 TO 7:POKE 1670+K,K*N:NEXT K
11050 POKE 1762,20:POKE 1764,20:POKE 1
766,100:POKE 1768,20
11060 N=INT((ALIENO-N1)/10):K=ALIENO-N
*10:X=(21-(K*N2))/N2:POSITION X,N+N1:F
OR Y=1 TO K:? #N6;"! ";:NEXT Y
11070 POSITION N1,N1:FOR Y=N1 TO ALIEN
O-K:? #N6;"! ";
11075 NEXT Y:POSITION N0,N0:? #N6;"sta
ge";STAGE;"      ":FOR K=N1 TO 200:NEXT
K:POSITION N0,N0:? #N6;"           "
11077 FOR T=N0 TO 4:POKE 1750+T,N0:POK
E 1755+T,N0:NEXT T
11080 FOR K=N0 TO 4:POKE 1771+K,N0:NEX
T K:POKE 53278,N0:POKE 1760,N0
11081 POKE 1767,50:POKE 77,N0
11082 POKE 712,0:POKE 711,56:POKE 710,
200:POKE 709,138:POKE 708,40
```

```
11090 A=SCRN+PEEK(1667):IF PEEK(A)>81
AND PEEK(A)<99 THEN POKE A,N0
11091 IF PEEK(A+20)>81 AND PEEK(A+20)<
99 THEN POKE A+20,N0
11093 FOR K=N0 TO 9:N=SCRN+PEEK(1716+K
):IF PEEK(N)>17 AND PEEK(N)<35 THEN PO
KE N,N0
11095 IF PEEK(N+20)>17 AND PEEK(N+20)<
35 THEN POKE N+20,N0
11097 NEXT K:POKE SCRN+PEEK(1669),N0:P
OKE SCRN+PEEK(1669)+1,N0
11098 FOR G=0 TO 4:IF PEEK(1651+G)>20
THEN A=(TOP-5+G)*256+PEEK(1651+G):FOR
K=A TO A-20 STEP -1:POKE K,N0:NEXT K
11099 NEXT G:FOR T=N0 TO 4:POKE 1651+T
,N0:NEXT T:POKE 1669,230:POKE 1667,210
12000 REM
12010 FOR K=N0 TO 9:POKE 1716+K,N0:NEX
T K:POKE 1726,9-BO
12011 POKE 1691,N1:POKE 1692,N1:POKE 1
778,N0:POKE 1760,N0
12016 TRAP 12020:WA=USR(USER):SOUND N0
,N0,N0,N0:SOUND N1,N0,N0,N0
12020 IF PEEK(1760)>N0 THEN 15010
12030 STAGE=STAGE+N1:GOTO 11010
15000 REM **PLAYER DIED
15010 SHIPS=SHIPS-N1:POSITION N0,N0:?
#N6;"SHIPS LEFT:";SHIPS;"  ";
15015 POKE SCRN+PEEK(1669),38:POKE SCR
N+PEEK(1669)+N1,39:FOR T=200 TO N0 STE
P -3:SOUND N0,T,6,10:NEXT T
15020 POKE SCRN+PEEK(1669),N0:POKE SCR
N+PEEK(1669)+N1,N0:SOUND N0,N0,N0,N0
15045 FOR K=N0 TO 4:IF PEEK(1651+K)>N0
 THEN POKE SCRN+PEEK(1681+K),PEEK(1701
+K)
15046 NEXT K
15048 POSITION N0,N0:? #N6;"
  ";
15049 SC=PEEK(SCRN+18)-16+(PEEK(SCRN+1
7)-16)*10+(PEEK(SCRN+16)-16)*100+(PEEK
(SCRN+15)-16)*1000
15050 FOR K=N0 TO 4:IF PEEK(1750+K)>N0
 THEN POKE 1750+K,N0:POKE 1770,PEEK(17
70)-N1
15051 NEXT K:POKE 1693,N0:IF PEEK(1729
)=1 AND PEEK(1739)<2 THEN POKE 1739,2
15053 SC=SC*10:IF SHIPS>N0 THEN 11080
15055 IF SC>HS THEN HS=SC
15060 POSITION N0,N0:? #N6;"     game
 over    ":FOR K=N1 TO 120:NEXT K:GOT
O 5410
```

CHOPPER MISSION

(24K CASSETTE, 32K DISK)

Hostages have been released from a plane half a mile in the air; an anti-gravity ray has fortunately been put in place to slow their descent, but there are other threats to their survival. On this rescue mission, at the controls of your chopper, you must rescue these falling unfortunates before four enemy helicopters and two enemy trucks destroy them.

POKE in the machine language and the title screen will appear on the screen. Press SELECT to change the stage you start on, and START to begin the game. Use a joystick in port 1 to move left and right. Press the fire button to go up. Ten points, multiplied by the stage, are awarded for every rescued man. The high score is displayed at the end of the game.

```
10 REM **CHOPPER MISSION
20 REM **BY Cliff McConnell
30 REM **ALL REMS MAY BE OMITTED
90 READ N0,N1,N2,N6,M,PHITE,PPOS,MPOSL
O,MPOSHI,CHITE,CPOS,DVFLAG,MSPEED,TSPE
ED,TPOS0,TPOS1,CSPEED,TIMER,SHCOLL,PA
120 TOP=PEEK(106):IVFLAG=1747:MCHAR=16
96:START=((TOP-16)*256)+230:POS=N2:DIM
 CH$(16):CH$="chopper mission "
125 DATA 0,1,2,6,0,208,209,1706,1716,1
729,1732,1741,1743,1751,1753,1754,1755
,1766,1760,256
130 DLI=START+1116:DVBLANK=START+42:IV
```

```
BLANK=START+850:DLI2=START+1144:USER=D
LI2+22
200 REM
220 MYPE=PEEK(106)-8:POKE M+N1,MYPE:PO
KE M,N0
225 POKE 106,MYPE+N2+N1:GRAPHICS 18:PO
SITION 4,N6:? #N6;"PLEASE WAIT":GOSUB
12010
230 A=MYPE*PA:FOR G=768 TO 2048:POKE A
+G,N0:NEXT G:POKE 712,224:POKE 708,62:
GOSUB 12010
240 FOR G=MPOSHI TO MPOSHI+9:POKE G,N0
:NEXT G
250 FOR G=CHITE TO CHITE+N2:POKE G,100
:NEXT G:POKE TPOS0,35:POKE TPOS1,210
260 FOR G=200 TO 207:READ N:POKE G+A+1
280,N:POKE G+A+1536,N:NEXT G:POKE 711,
26:GOSUB 12010
265 DATA 63,63,255,191,191,255,255,82
270 POKE 54279,MYPE:POKE 559,62:POKE 5
3277,3:POKE 705,120:POKE 706,204:POKE
707,46:POKE 709,70:POKE 710,212
275 GOSUB 12010
280 GOSUB 10010
310 GOSUB 11010
320 POKE IVFLAG,N1:POKE DVFLAG,N1:? #N
6;CHR$(125);:A=USR(START):POKE 1759,5:
POKE 1758,5
470 FOR G=N0 TO N2:POKE 53253+G,53+G*7
0:NEXT G:POKE 53252,123
500 REM **CHECK FOR START
505 POSITION N0,N0:? #N6;" PLEASE PRES
S START ":FOR K=N1 TO 50:IF PEEK(53279
)=N6 THEN 1020
520 NEXT K
530 POSITION N0,N0:? #N6;"score";SC;"
    ";:POSITION 11,N0:? #N6;"high";HS
;"  "
540 FOR K=N1 TO 200:IF PEEK(53279)=N6
THEN 1020
```

```
550 NEXT K:GOTO 505
1000 REM **START GAME
1020 FOR K=N0 TO 9:POKE MCHAR+K,N0:POK
E MPOSHI+K,N0:NEXT K
1030 CHOPPERS=N2+N1:SCREEN=PEEK(88)+25
6*PEEK(89):STAGE=N1:SC=N0
1035 FOR K=N0 TO 9:HI=INT((SCREEN+20+2
*K)/256):POKE 1772+K,(SCREEN+20+2*K)-H
I*256:POKE 1782+K,HI:NEXT K
1037 MLEFT=N0:POKE 1765,9
1038 FOR K=N0 TO 9:IF PEEK(1782+K)>0 T
HEN MLEFT=MLEFT+N1
1039 NEXT K:MSAVED=N0
1060 K=INT(STAGE/7):G=INT(STAGE/13):PO
KE MSPEED,STAGE*3:POKE TSPEED,4+N2*(ST
AGE-K*7):POKE CSPEED,2+STAGE-G*13
1065 IF PEEK(MSPEED)>24 THEN POKE MSPE
ED,24
1110 FUEL=1000+ABS(11-STAGE)*100
1140 POSITION N0,N0:? #N6;"       STAG
E ";STAGE;"        ";:SOUND N0,255,10,
8:SOUND 1,254,10,8
1141 FOR K=N0 TO N2:POKE CPOS+K,45:NEX
T K
1142 POKE PHITE,160:POKE PPOS,120:FOR
K=1024 TO 1280:POKE MYPE*PA+K,N0:NEXT
K:POKE 207,16
1143 POKE 53278,N0:FOR T=N0 TO N2+N2:P
OKE T+1760,N0:NEXT T
1145 ? #N6;CHR$(125);:POSITION N0,N0:?
 #N6;"score";SC:POSITION 12,N0:? #N6;"
fuel";FUEL
1146 COLOR 129:PLOT N0,11:DRAWTO 19,11
1147 FOR G=100 TO 103:POKE MYPE*PA+768
+G,252:NEXT G:POKE 53260,255
1148 FOR G=140 TO 143:POKE MYPE*PA+768
+G,3:NEXT G
1149 POKE 704,30:SOUND N1,N0,N0,N0
1150 SOUND N0,200,12,N2:A=USR(USER):GO
TO 2010+1000*PEEK(24)
```

```
2000 REM **FUEL OUT
2010 GOTO 5010
3000 REM **ENEMY HIT MAN
3010 N=0
3020 IF PEEK(53253+N)>N0 THEN 3050
3030 N=N+N1:IF N<N2+N1 THEN 3020
3040 GOTO 3091
3050 FLAG=N0:A=CPOS+N:B=CHITE+N:GOSUB
9010
3052 COL=INT(COL/2)
3055 K=PEEK(MPOSLO+COL)+PA*PEEK(MPOSHI
+COL)-SCREEN+20
3060 IF PEEK(MCHAR+COL)<>42 AND PEEK(M
POSHI+COL)<>0 THEN 3080
3070 A=TPOS0+N:GOSUB 9010:COL=INT(COL/
2)
3075 IF PEEK(MCHAR+COL)=42 THEN FLAG=N
1
3080 IF PEEK(MPOSHI+COL)=N0 THEN 3091
3081 POKE MCHAR+COL,N0:K=PEEK(MPOSLO+C
OL)+PA*PEEK(MPOSHI+COL):POKE MPOSHI+CO
L,N0:POKE K,N0:POKE MPOSLO+COL,N0
3082 POKE 1772+COL,N0:POKE 1782+COL,N0

3085 IF FLAG=N0 THEN POKE K+20,N0
3090 MLEFT=MLEFT-N1:FOR T=N0 TO 2000 S
TEP 80:SOUND N0,T,10,10:NEXT T
3091 POKE 53278,N0:FOR T=N0 TO 4:POKE
1760+T,N0:NEXT T
3092 SOUND N0,N0,N0,N0:IF MLEFT>N0 THE
N 3100
3095 IF MSAVED=N0 THEN CHOPPERS=N0:GOT
O 5030
3097 GOTO 7510
3100 GOTO 1150
4000 REM **PLAYER RESCUED MAN
4010 A=PPOS:GOSUB 9010
4015 COL=INT(COL/2):G=PEEK(MCHAR+COL)
4020 POKE MCHAR+COL,N0:K=PEEK(MPOSLO+C
OL)+PA*PEEK(MPOSHI+COL):POKE MPOSHI+CO
```

```
L,N0:POKE K,N0:POKE MPOSLO+COL,N0
4030 IF G<>42 THEN POKE K+20,N0
4040 MSAVED=MSAVED+N1:MLEFT=MLEFT-N1
4045 FOR T=255 TO N0 STEP -5:SOUND N0,
N0,N0,N0:SOUND N0,T,10,10:NEXT T
4050 IF MLEFT=N0 THEN 7510
4060 FOR G=N0 TO N2+N2:POKE SHCOLL+G,N
0:NEXT G:POKE 53278,N0
4070 GOTO 7010
5000 REM **PLAYER HIT ENEMY
5010 CHOPPERS=CHOPPERS-1
5011 POSITION N0,N0:? #N6;" CHOPPERS L
EFT:";CHOPPERS;"          "
5012 FOR T=10 TO N0 STEP -0.2:SOUND N0
,150-RND(0)*10*T,8,T:NEXT T
5020 IF CHOPPERS>0 THEN 1110
5030 POSITION N0,N0:? #N6;"      game
over     "
5035 IF SC>HS THEN HS=SC
5040 FOR T=N0 TO 400:NEXT T:GOTO 505
7000 REM
7010 SC=SC+10*STAGE:POSITION N0,N0:? #
N6;"score";SC
7020 GOTO 1150
7500 REM
7510 STAGE=STAGE+N1
7520 FUEL=(PEEK(SCREEN+16)-16)*100+(PE
EK(SCREEN+17)-16)*10+PEEK(SCREEN+18)-1
6
7530 SC=SC+INT(FUEL/10)*10
7540 POSITION 5,N0:? #N6;SC
7545 MLEFT=N0
7550 FOR G=0 TO 9:IF PEEK(1782+G)>N0 T
HEN MLEFT=MLEFT+N1
7560 NEXT G
7570 POSITION (20-MLEFT)/2,N6:FOR G=1
TO MLEFT:? #N6;"J";:NEXT G
7580 GOTO 1037
9000 REM
9010 COL=(PEEK(A)-40)/8
```

```
9020 HITE=INT((PEEK(B)-22)/16)
9030 RETURN
10000 REM **POKE IN MACHINE LANGUAGE
10010 N=START:RESTORE 32000
10015 FOR K=1 TO 4
10020 READ A:IF A<>-N1 THEN POKE N,A:N
=N+N1:GOTO 10020
10025 GOSUB 12010:NEXT K
10030 K=INT(DLI/PA):G=INT(DLI2/PA):POK
E DLI2+N2,K:POKE DLI+N2,G:POKE START+2
0,G
10035 POKE DLI2+N6+N1,DLI-K*PA:POKE DL
I+N6+N1,DLI2-G*PA:POKE START+25,DLI2-G
*PA:G=INT(DVBLANK/PA)
10040 K=INT(IVBLANK/PA):POKE START+N2+
N2,G:POKE START+13,K:POKE START+N6,DVB
LANK-G*PA
10050 POKE START+15,IVBLANK-K*PA:RESTO
RE 32100:N=N0
10060 READ A:IF A<>-N1 THEN POKE 1536+
N,A:N=N+N1:GOTO 10060
10070 DLIST=PEEK(560)+256*PEEK(561):PO
KE DLIST+N2+N1,PEEK(DLIST+N2+N1)+128:P
OKE DLIST+14,PEEK(DLIST+14)+128
10100 RETURN
11000 REM **POKE IN NEW CHARACTER SET
11010 CHBASE=MYPE-N2:A=CHBASE*256
11030 RESTORE 11100
11040 READ N:IF N<>-N1 THEN FOR G=N0 T
O 7:READ K:POKE A+N*8+G,K:NEXT G:GOTO
11040
11045 POKE 1757,CHBASE
11050 RETURN
11100 DATA 26,24,60,24,255,24,24,36,10
2
11110 DATA 34,0,0,0,0,0,0,0,0
11120 DATA 27,0,24,60,24,255,24,24,36
11130 DATA 35,102,0,0,0,0,0,0,0
11140 DATA 28,0,0,24,60,24,255,24,24
11150 DATA 36,36,102,0,0,0,0,0,0
```

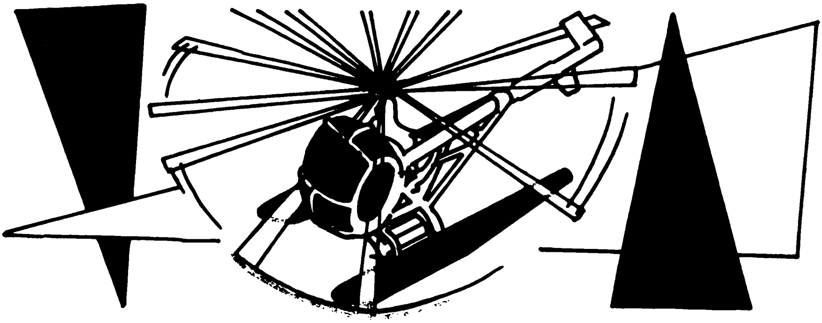
```
11160 DATA 29,0,0,0,24,60,24,255,24
11170 DATA 37,24,36,102,0,0,0,0,0
11180 DATA 30,0,0,0,0,24,60,24,255
11190 DATA 38,24,24,36,102,0,0,0,0
11200 DATA 31,0,0,0,0,0,24,60,24
11210 DATA 39,255,24,24,36,102,0,0,0
11220 DATA 32,0,0,0,0,0,0,24,60
11230 DATA 40,24,255,24,24,36,102,0,0
11240 DATA 33,0,0,0,0,0,0,0,24
11250 DATA 41,60,24,255,24,24,36,102,0

11270 DATA 42,24,60,24,255,24,24,36,10
2
11280 DATA 1,255,255,0,0,0,0,0,0
11290 DATA 0,0,0,0,0,0,0,0,0
11300 DATA -1
12000 REM **PRINT**CHOPPER MISSION
12010 FOR G=1 TO 2:POSITION POS,4:? #N
6;CH$(POS-1,POS-1);:POS=POS+N1
12020 SOUND N0,100,10,10:FOR T=N1 TO 1
0:NEXT T:SOUND N0,N0,N0,N0:NEXT G:RETU
RN
32000 DATA 104,169,7,162,128,160,42,32
,92,228,169,6,162,131,160,82,32,92,228
,169,132,141,1,2,169,120
32001 DATA 141,0,2,173,15,212,41,64,24
0,249,169,192,141,14,212,96,173,205,6,
240,3,76,98,228,165,207
32002 DATA 41,128,24,42,42,77,16,208,2
40,21,165,207,41,127,208,10,165,207,73
,128,133,207,230,207,208,21
32003 DATA 198,207,56,176,16,230,207,1
65,207,41,63,208,8,165,207,41,128,9,63
,133,207,173,16,208,208,68
32004 DATA 206,222,6,208,63,173,223,6,
141,222,6,165,88,24,105,16,133,205,165
,89,105,0,133,206,160,2
32005 DATA 177,205,24,136,113,205,136,
113,205,201,48,240,29,160,2,177,205,20
1,16,208,10,192,0,240,17,169
```

```
32006 DATA 25,145,205,208,8,56,233,1,1
45,205,24,144,3,136,16,229,165,207,41,
48,74,74,74,74,133,203
32007 DATA 165,207,41,15,24,109,206,6,
141,206,6,201,16,144,7,41,15,141,206,6
,230,203,165,207,48,9
32008 DATA 165,208,56,229,203,133,208,
208,60,173,8,208,24,109,9,208,109,10,2
08,109,11,208,41,1,208,7
32009 DATA 169,0,141,212,6,240,17,173,
212,6,208,21,173,213,6,208,7,169,1,141
,212,6,208,9,165,208
32010 DATA 24,101,203,133,208,208,10,1
65,207,16,4,169,128,133,207,165,208,20
1,42,176,8,169,0,133,207,169
32011 DATA 42,133,208,201,190,144,8,16
9,128,133,207,169,190,133,208,173,0,21
1,106,106,106,176,13,198,209,173
32012 DATA 210,6,41,2,141,210,6,24,144
,13,106,176,10,230,209,173,210,6,9,1,1
41,210,6,165,209,201,-1
32013 DATA 45,176,4,169,45,133,209,201
,200,144,4,169,200,133,209,162,2,173,2
19,6,74,74,74,133,203,173
32014 DATA 219,6,41,7,24,109,220,6,141
,220,6,201,8,144,7,41,7,141,220,6,230,
203,173,10,210,41
32015 DATA 63,208,34,173,10,210,41,15,
157,190,6,24,144,3,24,144,235,74,74,41
,1,240,6,29,199,6
32016 DATA 24,144,5,189,199,6,41,254,1
57,199,6,189,190,6,106,144,13,133,205,
189,193,6,56,229,203,157
32017 DATA 193,6,165,205,106,144,13,13
3,205,189,193,6,24,101,203,157,193,6,1
65,205,106,144,13,133,205,189
32018 DATA 196,6,24,101,203,157,196,6,
165,205,106,144,14,176,3,24,144,174,18
9,196,6,56,229,203,157,196
32019 DATA 6,189,193,6,201,42,176,13,1
```

```
89,190,6,73,1,157,190,6,169,42,157,193
,6,201,168,144,13,169
32020 DATA 168,157,193,6,189,190,6,73,
2,157,190,6,189,196,6,201,45,176,19,17
3,10,210,157,190,6,189
32021 DATA 199,6,9,1,157,199,6,169,45,
157,196,6,201,195,144,19,169,195,157,1
96,6,173,10,210,157,190
32022 DATA 6,189,199,6,41,2,157,199,6,
202,48,3,24,144,152,173,207,6,74,74,74
,74,133,203,173,207
32023 DATA 6,41,15,24,109,208,6,141,20
8,6,201,16,144,7,230,203,41,15,141,208
,6,166,203,240,50,238
32024 DATA 209,6,173,209,6,41,7,208,93
,141,209,6,160,9,185,160,6,41,63,201,4
2,240,76,185,180,6,-1
32025 DATA 240,71,132,204,133,206,185,
170,6,133,205,169,0,168,145,205,164,20
4,165,205,24,144,3,24,144,53
32026 DATA 105,20,153,170,6,165,206,10
5,0,153,180,6,185,170,6,24,105,20,133,
205,185,180,6,105,0,133
32027 DATA 206,160,0,177,205,240,12,16
4,204,185,160,6,41,192,9,42,153,160,6,
164,204,136,16,168,202,208
32028 DATA 150,173,209,6,24,105,26,133
,203,162,9,189,160,6,41,63,201,42,240,
10,189,160,6,41,192,5
32029 DATA 203,157,160,6,202,16,234,17
3,215,6,74,74,74,74,133,203,173,215,6,
24,109,216,6,141,216,6
32030 DATA 201,16,144,7,41,15,141,216,
6,230,203,173,217,6,56,229,203,141,217
,6,173,218,6,56,229,203
32031 DATA 141,218,6,173,8,208,24,109,
9,208,109,10,208,109,11,208,41,1,141,2
13,6,162,3,189,4,208
32032 DATA 157,224,6,202,16,247,173,12
,208,141,228,6,141,30,208,76,98,228,16
```

```
2,2,189,196,6,157,1,208
32033 DATA 202,16,247,173,211,6,240,3,
76,95,228,162,2,173,210,6,41,2,208,7,1
89,199,6,9,2,208
32034 DATA 5,189,199,6,41,253,157,199,
6,202,16,231,173,210,6,73,2,141,210,6,
168,169,0,192,0,240
32035 DATA 6,24,105,24,136,208,250,133
,203,170,165,1,24,105,4,133,206,169,0,
133,205,169,24,133,204,164
32036 DATA 208,189,0,6,145,205,200,232
,198,204,208,245,162,2,188,199,6,169,0
,192,0,240,6,24,105,24,-1
32037 DATA 136,208,250,157,202,6,202,1
6,235,174,202,6,169,24,133,204,172,193
,6,230,206,189,0,6,145,205
32038 DATA 200,232,198,204,208,245,174
,203,6,169,24,133,204,172,194,6,230,20
6,189,0,6,145,205,200,232,198
32039 DATA 204,208,245,174,204,6,169,2
4,133,204,172,195,6,230,206,189,0,6,14
5,205,200,232,198,204,208,245
32040 DATA 162,9,160,0,189,170,6,133,2
05,189,180,6,133,206,189,160,6,41,63,2
01,42,208,8,189,160,6
32041 DATA 145,205,24,144,26,189,160,6
,145,205,165,205,24,105,20,133,205,165
,206,105,0,133,206,189,160,6
32042 DATA 24,105,8,145,205,202,16,200
,165,209,141,0,208,238,231,6,208,3,238
,230,6,76,95,228,72,169
32043 DATA 132,141,1,2,169,120,141,0,2
,173,217,6,141,10,212,141,1,208,173,21
8,6,141,2,208,104,64
32044 DATA 72,169,132,141,1,2,169,92,1
41,0,2,173,221,6,141,10,212,141,9,212,
104,64,104,169,0,141
32045 DATA 205,6,141,211,6,165,88,24,1
05,16,133,24,165,89,105,0,133,25,160,2
,177,24,136,24,113,24
```

```
32046 DATA 136,113,24,201,48,208,4,169
,0,240,23,173,225,6,24,109,226,6,109,2
27,6,240,4,169,1,208
32047 DATA 7,173,224,6,240,19,169,2,23
8,205,6,172,231,6,204,231,6,240,251,23
8,211,6,133,24,96,173
32048 DATA 228,6,240,4,169,3,208,230,1
73,230,6,201,1,144,168,172,229,6,201,2
55,240,161,185,236,6,153
32049 DATA 170,6,169,0,153,160,6,185,2
46,6,153,180,6,169,0,141,230,6,206,229
,6,24,144,219,-1
32100 DATA 0,0,0,0,0,252,252,48,48,120
,120,127,255,250,250,254,32,80,0,0,0,0
,0,0,0,0
32101 DATA 0,0,0,63,63,12,12,30,30,254
,255,95,95,127,4,10,0,0,0,0,0,0,0,0,0,
0
32102 DATA 0,48,48,48,48,120,120,122,2
50,250,250,254,32,80,0,0,0,0,0,0,0,0,0
,0,0,12
32103 DATA 12,12,12,30,30,94,95,95,95,
127,4,10,0,0,0,0,0,0,0,255,255,255,2
55,255,255
32104 DATA 255,255,255,255,255,255,255
,255,0,0,0,0,0,0,0,-1
```

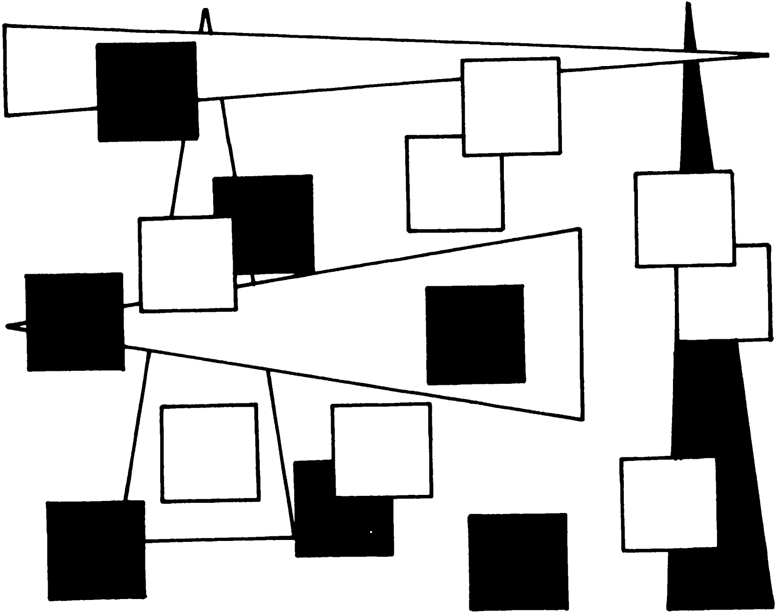


REVERSI

(16K CASSETTE, 24K DISK)

In this game of strategy, often called 'Othello', you must try to take over the board with your light blue pieces. You and the computer take it in turns to place pieces on the board until there are no free spaces remaining. To change the computer's pieces to your colour, you must move so that two of your pieces are on either side of one or more of the computer's. This can be either vertically, horizontally or diagonally.

Press SELECT to decide who has the first go, and START to begin the game. The board will be printed up and your cursor will appear. Use a joystick in port 1 to move the cursor to the square where you wish to place your piece, then press the fire button.



```
0 REM ******** REVERSI *********
1 REM *** by G. Ryan, 1984 ***
10 N0=0:N1=1:GOSUB 6000:DIM A(10,10)
20 FOR A=N1 TO 10:FOR B=N1 TO 10:IF A<
>N1 AND B<>N1 AND A<>10 AND B<>10 THEN
 A(A,B)=161:NEXT B:NEXT A
30 A(A,B)=131:NEXT B:NEXT A
40 CP=4:HP=36:SD=131:BL=161:HX=5:HY=5
50 A(5,5)=CP:A(6,6)=CP:A(5,6)=HP:A(6,5
)=HP:P=0:R=33
150 GOSUB 3000:IF W=N0 THEN GOTO 2000
999 REM COMPUTERS TURN
1000 POKE 77,N0:POSITION 1,11:? #6;"
           !MY TURN!"
1010 S=HP:T=CP:H=0
1020 FOR A=2 TO 9:FOR B=2 TO 9:IF A(A,
B)<>BL THEN 1170
1030 Q=0:FOR C=-1 TO 1:FOR D=-1 TO 1:K
=0:F=A:G=B
1040 IF A(F+C,G+D)<>S THEN 1060
1050 K=K+1:F=F+C:G=G+D:GOTO 1040
1060 IF A(F+C,G+D)<>T THEN 1080
1070 Q=Q+K
1080 NEXT D:NEXT C
1090 IF A=2 OR A=9 THEN Q=Q*2
1100 IF B=2 OR B=9 THEN Q=Q*2
1110 IF A=3 OR A=8 THEN Q=Q/2
1120 IF B=3 OR B=8 THEN Q=Q/2
1130 IF (A=2 OR A=9) AND (B=3 OR B=8)
THEN Q=Q/2
1140 IF (A=3 OR A=8) AND (B=2 OR B=9)
THEN Q=Q/2
1150 IF Q<H OR Q=0 OR (RND(0)>0.3 AND
Q=H) THEN 1170
1160 H=Q:M=A:N=B:HX=A-1:HY=B-1
1170 NEXT B:NEXT A
1180 IF H=0 AND R=0 THEN 5000
1190 IF H=0 THEN 1210
1200 GOSUB 4000
1210 Z=TP:GOSUB 3000:IF Z=TP THEN 5000
```

```
1999 REM PLAYERS TURN
2000 POKE 77,N0:POSITION 1,11:? #6;"!Y
OUR TURN!                ":GOSUB 8500
2010 GOSUB 8000:IF L=0 THEN GOSUB 8500
:GOTO 2010
2020 S=CP:T=HP
2030 GOSUB 4000
2040 GOSUB 3000
2050 GOTO 1000
2999 REM ADD UP PIECES
3000 C=0:H=0
3010 FOR B=2 TO 9:FOR A=2 TO 9:POSITIO
N 4+A,B-1:? #6;CHR$(A(A,B));:C=C+(A(A,
B)=CP):H=H+(A(A,B)=HP):TP=C+H
3020 NEXT A:NEXT B
3030 POSITION 9,10:? #6;H;"  ":POSITION
 18,10:? #6;C;"  "
3040 IF C+H>=64 THEN POP :GOTO 5000
3050 RETURN
3999 REM PLACE PIECE ON BOARD
4000 FOR C=-1 TO 1:FOR D=-1 TO 1:F=M:G
=N
4010 IF A(F+C,G+D)<>S THEN 4030
4020 F=F+C:G=G+D:GOTO 4010
4030 IF A(F+C,G+D)<>T THEN 4060
4040 A(F,G)=T:IF M=F AND N=G THEN 4060

4050 F=F-C:G=G-D:GOTO 4040
4060 NEXT D:NEXT C
4070 RETURN
4999 REM GAME OVER
5000 H=0:C=0:FOR B=2 TO 9:FOR A=2 TO 9
:H=H+(A(B,A)=HP):C=C+(A(B,A)=CP)
5010 NEXT A:NEXT B
5020 POSITION 9,10:? #6;H;"  ";:POSITIO
N 18,10:? #6;C;"  ";
5030 X=INT(H/(C+H)*100+0.5):POSITION 1
,11:? #6;" your pieces ";X;"%";"     ";
5040 POSITION 1,2:? #6;"GAME":POSITION
 1,4:? #6;"OVER"
```

```
5050 POSITION 15,2:? #6;"press":POSITI
ON 15,4:? #6;"start"
5060 IF PEEK(53279)<>6 THEN 5060
5070 PF=1:GOSUB 6100:GOTO 20
5999 REM INITIALIZATION
6000 RT=PEEK(740)-4:POKE 106,RT-2:GRAP
HICS 18:POKE 756,RT:RT=RT*256:DIM SD$(
10),BD$(10)
6010 POSITION 7,4:? #6;"reversi":POSIT
ION 1,6:? #6;"WRITTEN BY G. RYAN":FOR
A=1 TO 10:SD$(A)=CHR$(131)
6020 BD$(A)=CHR$(161):NEXT A:FOR A=RT
TO RT+7:POKE A,0:NEXT A
6030 FOR A=40 TO 47:POKE RT+A,PEEK(573
44+A):NEXT A
6040 FOR A=112 TO 511:POKE RT+A,PEEK(5
7344+A):NEXT A:RESTORE 6060
6050 READ I:IF I<>-1 THEN FOR A=I TO I
+7:READ D:POKE RT+A,D:NEXT A:GOTO 6050

6060 DATA 32,0,60,126,126,126,126,60,0

6070 DATA 8,0,126,126,126,126,126,126,
0
6080 DATA 24,170,85,170,85,170,85,170,
85
6090 DATA 72,255,129,129,129,129,129,1
29,255
6100 DATA 96,255,195,129,129,129,129,1
95,255,-1
6110 PUT #6,125:POSITION 5,0:? #6;SD$:
POKE 708,126:POKE 709,56:POKE 710,164:
POKE 711,70
6120 POSITION 5,9:? #6;SD$:FOR A=1 TO
8:POSITION 5,A:? #6;SD$(10);BD$(3);SD$
(10):NEXT A
6130 POSITION 1,10:? #6;"you own 0":PO
SITION 12,10:? #6;"i own 0"
6140 POSITION 1,11:? #6;"select:COMP G
O 1ST";:W=1
```

```
6150 IF PEEK(53279)=5 THEN 6180
6160 IF PEEK(53279)=6 THEN 6220
6170 GOTO 6150
6180 W=(W=0):IF W THEN POSITION 8,11:?
 #6;"COMP GO 1ST";:GOTO 6200
6190 POSITION 8,11:? #6;" YOU GO 1ST";
6200 IF PEEK(53279)<>7 THEN 6200
6210 GOTO 6150
6220 POSITION 1,11:? #6;"
     ";:RETURN
7999 REM INPUT PLAYERS MOVE WITH JOYST
ICK
8000 LOCATE 5+HX,HY,CH:HH=HX:HV=HY
8010 COLOR CH+8:PLOT 5+HX,HY:ST=STICK(
0):IF STRIG(0)=0 THEN 8050
8015 IF PEEK(53279)=1 THEN POP :COLOR
CH:PLOT HX+5,HY:GOTO 5000
8020 HX=HX+PTRIG(0)-PTRIG(1):HY=HY+(ST
=13 OR ST=5 OR ST=9)-(ST=14 OR ST=10 O
R ST=6)
8030 HX=HX+(HX<1)-(HX>8):HY=HY+(HY<1)-
(HY>8):IF HX<>HH OR HY<>HV THEN COLOR
CH:PLOT 5+HH,HV:GOTO 8000
8040 GOTO 8010
8050 IF CH<>161 THEN GOSUB 8500:GOTO 8
010
8060 COLOR CH:PLOT 5+HH,HV
8070 M=HX+1:N=HY+1:L=0:X=HX+5:Y=HY
8080 FOR A=-1 TO 1:FOR B=-1 TO 1
8090 LOCATE X+A,Y+B,C:IF C=4 THEN 8150
8100 NEXT B:NEXT A:RETURN
8150 H=X+A+A:V=Y+B+B
8160 LOCATE H,V,Z:IF Z=HP THEN L=N1:PO
P :POP :RETURN
8170 IF Z=131 OR Z=161 THEN 8100
8180 H=H+A:V=V+B:GOTO 8160
8500 SOUND N0,15,12,4:FOR Z=N1 TO 20:N
EXT Z
8510 IF STRIG(N0)=N0 THEN 8510
8520 SOUND N0,N0,N0,N0:RETURN
```

MARS LANDER

(16K CASSETTE, 24K DISK)

While flying past Mars, you suddenly notice the low reading on your fuel gauge. Since you don't have enough fuel left to completely escape the gravity of the planet, you must make a risky descent on an old landing pad on the surface of Mars. Before you can land you must avoid asteroids of all sizes, small particles of space dust and a rotating force-shield. Moreover, gravity constantly exerts its force on your ship and, in order to counteract its pull and land softly on the planet, you must use up fuel.

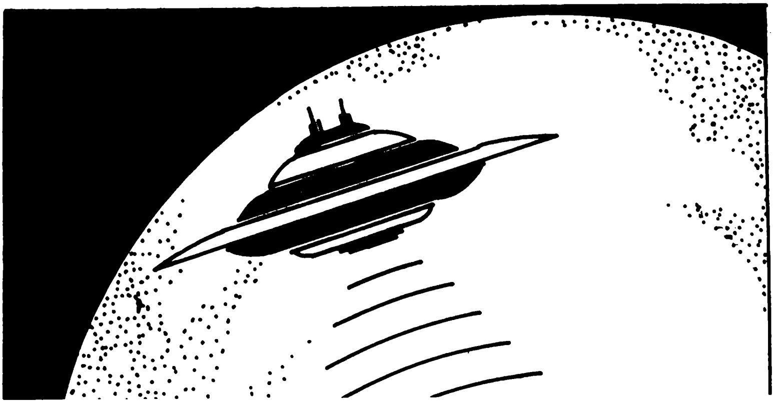
Press START to begin the game. Use a joystick in port 1 to move your craft left and right. Push the joystick down to make your ship descend faster than gravity can take it. Press the fire button to give your ship upward thrust against gravity.

```
10 REM **MARS LANDER
20 REM **By Cliff McConnell
30 REM **ALL REMS MAY BE OMITTED
120 POKE 704,0
130 HS=0:GOSUB 10010:GOSUB 12010
140 GOSUB 15010:? #6;CHR$(125):GOSUB 1
2010:GOSUB 13010:GOSUB 14010:Y=IN+37:X
=124:STAGE=1
143 POKE 53760,150:NO=0:L=3:SC=0:POKE
656,0:POKE 657,7:? "      ";
144 POKE 53270,0:W=20-STAGE*2:GOSUB 16
010:IF W<8 THEN W=8
145 COLOR 1:PLOT 0,50:DRAWTO 159,50:SH
=20:COLOR 0:PLOT 30,50:DRAWTO 30-W,50
148 Y=IN+37:X=124:U=35-RND(0)*10:G=30:
PM$(1)=CHR$(0):PM$(2304)=CHR$(0):PM$(2
)=PM$:F=500
```

```
149 POKE 53760,150:POKE 656,0:POKE 704
,250:TRAP 710:POKE 53278,0
150 V=STICK(0):G=G+V(V):Y=Y+G(G):U=U+H
(V):X=X+G(U):PM$(Y)=SH$:POKE 53248,X:F
=F+F(V):POKE 657,17:? F;" ";
155 COLOR 0:PLOT SH,50:COLOR 1:PLOT SH
-W,50:SH=SH+1
160 POKE 53761,N(V):IF F AND  NOT PEEK
(53252) THEN 150
165 POKE 53761,0:POKE 77,0
170 V=PEEK(53252):IF V>4 THEN 210
180 IF V>3 THEN 2010:REM BONUS
190 IF V>1 AND G<35 THEN 1010:REM LAND
ED
200 REM **DIED
210 PM$(Y)=EXPL$:FOR T=0 TO 255 STEP 5
:SOUND 0,T,10,10:POKE 704,(255-T)/16:N
EXT T
220 FOR T=255 TO 0 STEP -7:SOUND 0,T,1
0,10:NEXT T:SOUND 0,0,0,0
230 L=L-1:POKE 656,1:POKE 657,18:? L;
240 IF L>0 THEN 148
510 IF SC>HS THEN HS=SC
515 GOTO 140
520 RUN
710 COLOR 1:PLOT (SH-W)*(SH>W),50:SH=(
SH+1)*(SH<159+W):TRAP 710:GOTO 160
1000 REM **LANDED
1010 POP :STAGE=STAGE+1:COLOR 0:IF STA
GE/3=INT(STAGE/3) THEN COLOR 1
1020 PLOT 25,62:DRAWTO 135,62:POKE 656
,0
1045 POKE 656,1:POKE 657,7:? STAGE;
1050 SOUND 0,121,10,8:SOUND 1,120,10,8
:FOR K=0 TO 30:NEXT K:SOUND 0,0,0,0:SO
UND 1,0,0,0:FOR K=0 TO 5:NEXT K
1060 SOUND 0,121,10,8:SOUND 1,120,10,8
:FOR K=1 TO 200:NEXT K:SOUND 0,0,0,0:S
OUND 1,0,0,0:POKE 656,0
1080 FOR K=1 TO F STEP 5:F=(F-5)*(F-5)
```

```
0):SC=SC+1:POKE 657,7:? INT(SC);:POKE
657,17:? F;"  ";:SOUND 0,100,10,10
1090 FOR T=0 TO 10:NEXT T:SOUND 0,0,0,
0:NEXT K:SC=INT(SC)
1950 GOTO 144
2000 REM **GOT BONUS
2010 FOR T=0 TO NO
2011 IF  NOT BY(T) THEN NEXT T
2013 COLOR 0:PLOT BX(T),BY(T):POKE 532
78,0
2020 FOR K=200 TO 0 STEP -10:SOUND 0,K
,10,10:NEXT K:POKE 53760,150:V=PEEK(53
252)
2030 IF V<4 THEN SC=SC+STAGE*10:POKE 6
56,0:POKE 657,7:? SC;:BX(T)=0:BY(T)=0:
GOTO 150
2040 COLOR 3:PLOT BX(T),BY(T):NEXT T:G
OTO 150
10000 REM **SET UP PM GRAPHICS
10010 GRAPHICS 7:DL=PEEK(560)+256*PEEK
(561):POKE DL+85,70:POKE DL+88,6:POKE
DL+89,6:POKE DL+90,6:POKE 752,1
10030 Y=0:X=0:DIM PM$(2304),SH$(11),V(
15),H(15),G(80),F(15),N(15),BX(11),BY(
11),EXPL$(11)
10130 A=ADR(PM$):PM=INT(A/2048)*2048:X
=PM+1024:IF A>X THEN PM=PM+2048:X=X+20
48
10140 IN=X-A:POKE 710,158:POKE 708,52
10150 POKE 54279,PM/256:POKE 559,62:PO
KE 53277,3:POKE 53248,150
10200 FOR K=1 TO 11:READ V:SH$(K)=CHR$
(V):NEXT K
10210 DATA 0,0,0,16,16,56,124,40,0,0,0

10220 FOR K=0 TO 15:READ V:N(K)=V:NEXT
 K
10230 DATA 0,0,0,0,0,4,134,4,0,4,134,4
,0,0,134,0
10240 FOR K=1 TO 11:READ V:EXPL$(K,K)=
CHR$(V):NEXT K
10250 DATA 0,73,42,28,8,127,8,28,42,73
,0
11000 REM **JOYSTICK VARIABLES
11010 RESTORE 11020:FOR K=0 TO 15:READ
 V,H:V(K)=V:H(K)=H:NEXT K
11020 DATA 0,0,0,0,0,0,0,0,0,0,.4,.2,-
.3,.2,.2,.2,0,0,.2,-.2,-.3,-.2,.2,-.2,
0,0,.4,0,-.3,0,.4,0
11030 FOR K=0 TO 60:G(K)=(K-30)*0.1:NE
XT K:FOR K=61 TO 80:G(K)=3:NEXT K
11040 FOR K=0 TO 15:F(K)=0:NEXT K:F(14
)=-1:F(10)=-1:F(6)=-1
11050 RETURN
12000 REM **DRAW SCREEN
12010 COLOR 1:PLOT 0,0:DRAWTO 159,0:DR
AWTO 159,79:DRAWTO 0,79:DRAWTO 0,0
12015 PLOT 0,79:T=0
12020 X=INT(RND(0)*5):T=T+INT(RND(0)*5
):IF T<159.5 THEN DRAWTO T,79-X:GOTO 1
2020
12990 RETURN
13000 REM **SET UP OBSTACLES
13010 FOR T=0 TO 5:PLOT RND(0)*158,RND
(0)*12+52:NEXT T
13020 COLOR 2:PLOT 77,74:DRAWTO 83,74
13030 COLOR 1:PLOT 84,70:DRAWTO 84,75:
DRAWTO 76,75:DRAWTO 76,70
13050 RETURN
14000 REM **PRINT SCORE AND FUEL
14010 SOUND 0,0,0,0
14015 POKE 656,0:POKE 657,21:? "HIGH S
CORE:";HS;
14020 POKE 657,1:? "SCORE:";SC;"      ";
:POKE 657,12:? "FUEL:";F;
14050 POKE 656,1:POKE 657,1:? "STAGE:1
          ";:POKE 657,12:? "SHIPS:3";
14220 RETURN
15000 REM **WAIT FOR START
```

```
15010 POKE 656,1:POKE 657,1:? "please
press start";
15015 IF HS>0 THEN POKE 656,0:POKE 657
,32:? HS;
15018 POKE 656,1:POKE 657,21:? "MARS L
ANDER";
15020 IF PEEK(53279)<>6 AND STRIG(0)<>
0 THEN 15020
15030 RETURN
16000 REM **PUT UP EXTRA OBSTACLES
16010 COLOR 1:FOR T=1 TO 3:PLOT RND(0)
*158,RND(0)*12+52:NEXT T
16020 N=RND(0)*4:FOR T=1 TO N:S=INT(RN
D(0)*3)+1:H=RND(0)*(157-10*S)+3*S:V=RN
D(0)*(39-7*S)+10
16030 PLOT H+5*S,V:RESTORE 16200:FOR K
=1 TO 23:READ X,Y:DRAWTO H+X*S,V+Y*S
16035 NEXT K:NEXT T
16040 COLOR 0:FOR T=0 TO NO:PLOT BX(T)
,BY(T):NEXT T
16050 COLOR 3:NO=NO+(NO<10)*1:FOR T=0
TO NO:BX=RND(0)*158:BY=RND(0)*40+20:BX
(T)=BX:BY(T)=BY:PLOT BX,BY:NEXT T
16080 RETURN
16200 DATA 3,0,3,1,2,1,2,2,1,2,1,4,0,4
,0,5,2,5,2,5,2,6,4,6,4,7,5,7,5,6,6,6,6
,4,7,4,7,3,6,3,6,2,5,2,5,0
```



How To Write Better Programs

INVENTING AND DEVELOPING YOUR OWN GAMES PROGRAMS

By Series Editor, Tim Hartnell

It's all very well spending your time typing in programs like those in this book, but there is sure to come a time when you decide you'd like to develop some games programs of your own. In this section of the book, I'd like to discuss a few ideas which may help you write games which you'll both enjoy developing and — more importantly — you and your friends will enjoy playing.

HAVE A CLEAR GOAL IN MIND

Although in many (perhaps most) cases, your computer program will take on a life of its own as you write it, developing away from the concept you had in mind when you started programming, it is important at the outset to have a pretty good idea of what your game will involve.

This is not as obvious a suggestion as you might think. Of course, you'll know if you're developing a 'chase the ghosts around the maze as you eat the power pills' program that you are going to need a different sort of program layout to one which places you inside a Haunted Oak, peopled with gremlins and halflings. But you have to go beyond the basic "I'm going to write me an Adventure" stage to work out such things as (a) what the object of the game will be; (b) what the screen display will look like; (c) what variables, and variable names, you'll need;

(d) the nature of the player input; (e) how 'winning' or 'losing' will be determined; and so on.

Let's look at these one by one.

THE OBJECT OF THE GAME

This can usually be stated very succinctly: "To find the lost treasure of the Aztecs"; "To destroy as many asteroids as possible before running out of ships"; or "To play a game of chess". But even though this stage of the game production can be accomplished very quickly, it should not be overlooked. Get this statement — which might be just a sentence, or may run to a paragraph length or more, if there is more than one 'screen' to be worked through, with a different scenario for each screen — down in writing.

You may well discard the original aim as the program develops, and it looks like the direction it is taking is better than the one you first thought of. Despite this, it is important to have something concrete to aim at, to stop you wasting hour after hour doodling aimlessly.

THE SCREEN DISPLAY

I've found that making a sketch, or sketches, of what the display will look like once the program is up and running, is of tremendous benefit. Once you have your drawing, and it doesn't matter how rough it is so long as it shows all the important things you want on the screen, and their relative positions and size, you'll discover the program concept is likely to crystalize.

As well as seeing immediately how you will write parts of the code to achieve the game's aim, you'll get an idea of whether or not the game is even worth writing in the form you had considered. Perhaps the game will be too complex if you leave everything on the screen you were intending to; or maybe most of the screen will be wasted space, so that a more complicated game scenario should be devised.

I've discovered that sketching the proposed screen display before starting to program is particularly useful, especially when creating arcade and simulation games. You get an indication of the variables you'll need, the user-defined graphics, the kind of player inputs which will be most conducive to good player interaction, and so on.

Simulation games, as you probably know, are those in which the computer models an external reality — such as running a cake shop, a war, or an airport — and allows you to experience (after a fashion) what it would be like to take part in such an activity in real life. Simulation games are not particularly difficult to write — in terms of needing clever coding — but instead demand a methodical, painstaking approach to the program.

In my book *The ZX Spectrum Explored* (Sinclair Browne, 1982), there is a program with the unlikely name of 'Workin' for the Man', in which you are running a factory, staffed with a highly-erratic workforce, involved in the manufacture of some mythical product called 'The Zibby'. The player gets a factory report two or three times a week, and from this report has to decide how many staff he or she will hire or (attempt to) fire, how many Zibbies will be the production target for the week, and so on.

This report is the key to the program, and when I wrote the game, I started by making a sketch of how the screen would look. It was a bit like this:

```
FACTORY REPORT: WEEK 5

Capital in hand is $2,657.92


Your stores hold 12 Zibbies worth $169.68
They sell for $14.14 each and cost $7.41
each to make


Workforce is 7 people
Their wages are $41 each and the wage bill
this week is $287
```

Each person can make 10 Zibbies a week,
a total output of 70

Once I had this sketch drawn up, I was ready to go. As you can see, it gives a very good indication of the variables which will be needed. For a start, I know I'll have to control the number of the week, the capital, the contents of the stores (and their value) and so on.

I found that once I'd completed the screen display sketch, the rest of the program was relatively easy to write. Doing a sketch in this way gives you an instant guide to the main variables you'll need.

USE HELPFUL VARIABLE NAMES

I also tend to use variable names which relate in some way to that which they are representing, as it saves having to keep a list of the variables which have been assigned, and what they've been assigned to. For example, I could use WK for week, CH for capital in hand, MZ for the cost of making each Zibby and SZ for the selling price. If Z was the number of Zibbies, I would know that the total value of Zibbies I had was Z (the number of them) multiplied by SZ (their selling price) and it cost me Z multiplied by MZ (their price of manufacture) to make them. My profit, if I sold them all, would then be Z*SZ minus Z*MZ.

If you follow a similar idea, you'll find it is much easier to keep track of what is happening in your program than might otherwise be the case.

THE NATURE OF THE PLAYER INPUT

It's important to make games *easy* and fun to play. It's not good having the best Asteroids-derivative program in the world if players have trouble hitting the fire button because you've placed it right next door to the 'rotate' control.

Many programs which provide 'up', 'down', 'right' and 'left' controls, automatically use arrow or cursor keys, even though these might be most inconvenient for the player to use. Have a look at your keyboard, and see if you can find better ones. I often use "Z" and "M" for programs which need just left and right movement, with the space bar for fire. These keys seem logical to me, and no player time is wasted in learning them, or trying to remember them when the game is underway. In a similar way, I tend to use "A" (for up) and "Z" (for down) for the left hand, and the "greater than" and "less than" keys for left and right (pointing out to the player that the < and > symbols point in the relevant directions).

Use INKEY$ or GET$ whenever you can, to prevent the player from having to use the RETURN or ENTER keys to get the program underway.

HOW THE GAME WILL END

The way the game will be won and lost needs to be defined, and clear to the player. Do you need to blast all the aliens to win, and will you lose automatically if one alien lands, and you've still got ships left, or only if you have no ships left. In a two-player game, is the loser the first player to lose three lives, or seven pieces, or does the game only end when the *difference* between the two scores is three or seven or whatever.

Work this out, and make it very clear to the player. Whether the goal of the game is to clear the left-hand side of the screen of the Screaming Widgies, or to clock up a fortune of $7.3 billion, it must be both clear to the player, and *possible to achieve*. A 'win condition' which can never be achieved on the higher levels of play is most unsatisfactory. No matter how difficult it is to do, you are only defrauding players if you set goals whose achievement is not possible within the constrictions you've put into the game.

I hope these five points may give you a few ideas on how you can go ahead and write programs which will be relatively easy to write, and which will be satisfying for you and your friends to play.

GLOSSARY

A

Accumulator — the place within the computer in which arithmetic computations are performed and where the results of these computations are stored.

Algorithm — the series of steps the computer follows to solve a particular problem.

Alphanumeric — this term is usually used in relation to a keyboard, as in 'it is an alphanumeric keyboard', which means that the keyboard has letters as well as numbers. It is also used to refer to the 'character set' of the computer. The character set comprises the numbers and letters the computer can print on the screen.

ALU (Arithmetic/Logic Unit) — the part of the computer which does arithmetic (such as addition, subtraction) and where decisions are made.

AND — a Boolean logic operation that the computer uses in its decision-making process. It is based on Boolean algebra, a system developed by mathematician George Boole (1815-64). In Boolean algebra the variables of an expression represent a logical operation such as OR and NOR.

ASCII — stands for American Standard Code for Information Exchange, the most widely used encoding system for English language alphanumerics. There are 128 upper and lower case letters, digits and some special characters. ASCII converts the symbols and control instructions into seven-bit binary combinations.

Assembler — a program which converts other programs written in assembly language into machine code (which the computer can understand directly). Assembly language is a low level programming language which uses easily memorised combinations of two or three letters to represent a particular instruction which the assembler then converts so the machine can understand it. Examples of these are ADD (add), and SUB (subtract). A computer programmed in assembly language tends to work more quickly than one programmed in a higher level language such as BASIC.

B

BASIC — an acronym for Beginners All-Purpose Symbolic Instruction Code. It is the most widely used computer language in the microcomputer field. Although it has been criticised by many people, it has the virtue of being very easy to learn. A great number of BASIC statements resemble ordinary English.

Baud — named after Baudot, a pioneer of telegraphic communications. Baud measures the rate of transfer of information and is approximately equal to one bit per second.

BCD — an abbreviation for Binary Coded Decimal.

Benchmark — a test against which certain functions of the computer can be measured. There are a number of so-called 'standard Benchmark tests', but generally these only test speed. This is rarely the aspect of a microcomputer that is most of interest to the potential buyer.

Binary — a numbering system that uses only zeros and ones.

Bit — an abbreviation for Binary Digit. This is the smallest unit of information a computer circuit can recognise.

Boolean Algebra — the system of algebra developed by mathematician George Boole which uses algebraic notation to express logical relationships (see AND).

Bootstrap — a short program or routine which is read into the computer when it is first turned on. It orients the computer to accept the longer, following program.

Bug — an error in a computer program which stops the program from running properly. Although it is generally used to mean only a fault or an error in a program, the term bug can also be used for a fault in the computer hardware.

Bus — a number of conductors used for transmitting signals such as data instructions, or power in and out of a computer.

Byte — a group of binary digits which make up a computer word. Eight is the most usual number of bits in a byte.

C

CAI — Computer Assisted Instruction.

CAL — Computer Assisted Learning. The term is generally used to describe programs which involve the learner with the learning process.

Chip — the general term for the entire circuit which is etched onto a small piece of silicon. The chip is, of course, at the heart of the microcomputer.

Clock — the timing device within the computer that synchronises its operations.

COBOL — a high level language derived from the words Common Business Orientated Language. COBOL is designed primarily for filing and record-keeping.

Comparator — a device which compares two things and produces a signal related to the difference between the two.

Compiler — a computer program that converts high level programming language into binary machine code so the computer can handle it.

Complement — a number which is derived from another according to specified rules.

Computer — a device with three main abilities or functions:
1) to accept data
2) to solve problems
3) to supply results

CPU — stands for Central Processing Unit. This is the heart of the computer's intelligence, where data is handled and instructions are carried out.

Cursor — a character which appears on the TV screen when the computer is operating. It shows where the next character will be printed. On a computer there are usually 'cursor control keys' to allow the user to move the cursor around the screen.

D

Data — information in a form which the computer can process.

Debug — the general term for going through a program and correcting any errors in it, that is, chasing down and removing bugs (see Bug).

Digital Computer —a computer which operates on information which is in a discrete form.

Disk/Disc — this is a magnetically sensitised plastic disk, a little smaller than a single play record. This is used for storing programs and for obtaining data. Disks are considerably faster to load than a cassette of the same length program. The disk can be searched very quickly while a program is running for additional data.

Display — the visual output of the computer, generally on a TV or monitor screen.

Dot Matrix Printer — a printer which prints either the listing of a program or that which is displayed on the TV screen. Each letter and character is made up of a number of dots. The higher the number of dots per character the finer the resolution of the printer.

Dynamic Memory — a memory unit within the computer which 'forgets' its contents when the power is turned off.

E

Editor — this term is generally used for the routine within the computer which allows you to change lines of a program while you are writing it.

EPROM — stands for Erasable Programmable Read-Only Memory. This is like the ROM in the computer, except that it is fairly easy to load material into an EPROM and it doesn't disappear when you turn the power off. EPROMs must be placed in a strong ultra violet light to erase them.

Error Messages — the information given by a computer where there is a fault in the coding during a part of a program, usually shown by the computer stopping, and printing a word, or a word and numbers, or a combination of numbers only, at the bottom of the screen. This tells you what mistake has been made. Common mistakes include using the letter O instead of zero in a line, or leaving out a pair of brackets, or one of the brackets, in an expression, or failing to define a variable.

F

File — a collection of related items of information organised in a systematic way.

Floppy Disk — a relatively cheap form of magnetic disk used for storing computer information, and so named because it is quite flexible (see Disk/Disc).

Flow Chart — a diagram drawn up before writing a program, in which the main operations are enclosed within rectangles or other shapes and connected by lines, with arrows to represent loops, and decisions written at the branches. It makes writing a program much easier because traps such as infinite loops, or non-defined variables can be caught at an early stage. It may not be worth writing a flow chart for very short programs, but generally a flow chart aids in creating programs.

Firmware — there are three kinds of 'ware' in computers: software 'temporary' programs; hardware like the ROM which contains permanent information; and firmware in which the information is relatively permanent, as in an EPROM (see EPROM).

Flip-Flop — a circuit which maintains one electrical condition until changed to the opposite condition by an input signal.

FORTRAN — an acronym for FORmula TRANslation, this is a high level, problem orientated computer language for scientific and mathematical use.

G

Gate — an electrical circuit which, although it may accept one or more incoming signals, only sends out a single signal.

Graphics — pictorial information as opposed to letters and numbers.

H

Hard Copy — computer output which is in permanent form.

Hardware — the physical parts of the computer (also see software and firmware).

Hexadecimal (Hex) — a numbering system to the base sixteen. The digits zero to nine are used, as well as the letters A, B, C, D, E and F to represent numbers. A equals 10, B equals 11, C equals 12, and so on. Hex is often used by microprocessor users.

Hex Pad — a keyboard designed specifically for entering hexadecimal notation.

High Level Language — a programming language which allows the user to talk to the computer more or less in English. In general, the higher the level of the language (that is, the closer it is to English), the longer it takes for the computer to translate it into a language it can use. Lower level languages are far more difficult for human operators but are generally executed far more quickly.

I

Input — the information fed into the computer via a keyboard, a microphone, a cassette or a disk.

Input/Output (I/O Device) — a device which accepts information or instructions from the outside world, relays it to the computer, and then, after processing, sends the information out in a form suitable for storing, or in a form which could be understood by a human being.

Instruction — data which directs a single step in the processing of information by the computer (also known as a command).

Integrated Circuit — a complete electronic circuit imprinted on a semiconductor surface.

Interface — the boundary between the computer and a peripheral such as a printer.

Interpreter — a program which translates the high level language fed in by the human operator, into a language which the machine can understand.

Inverter — a logic gate that changes the signal being fed in, to the opposite one.

Interactive Routine — part of a program which is repeated over and over again until a specified condition is reached.

J

Jump Instruction — an instruction which tells the computer to go to another part of the program, when the destination of this move depends on the result of a calculation just performed.

K

K — this relates to the size of the memory. Memory is usually measured in 4K blocks. 1K contains 1,024 bytes.

Keyword — the trigger word in a line of programming, usually the first word after the line number. Keywords include STOP, PRINT and GOTO.

L

Language — computer languages are divided into three sections: high level languages, such as BASIC, which are reasonably close to English and fairly easy for humans to use; low level languages, such as Assembler, that use short phrases which have some connection with English (ADD for add and RET for return, for instance); and machine code which communicates more or less directly with the machine.

LCD — this stands for Liquid Crystal Diode. Some computers such as the TRS-80 Pocket Computer use an LCD display.

LED — this stands for Light Emitting Diode. The bright red numbers which are often used on watch or clock displays are made up of LEDs.

Logic — the mathematical form of a study of relationships between events.

Loop — a sequence of instructions within a program which is performed over and over again until a particular condition is satisfied.

M

Machine Language or Machine Code — an operation code which can be understood and acted upon directly by the computer.

Magnetic Disk — see Disk and Floppy Disk.

Mainframe — computers are generally divided into three groups, and the group a computer falls into depends more or less on its size. The computer you are thinking of buying is a microcomputer; medium sized computers are known as minicomputers; and the giant computers that you sometimes see in science fiction movies are mainframe computers. Until 15 years ago mainframe computers were, in practical terms, the only ones available.

Memory — there are two types of memory within a computer. The first is called ROM (read-only memory); this is the memory that comes already programmed on the

Numeric — pertaining to numbers as opposed to letters (that is, alphabetic). Many keyboards are described as being alphanumeric which means both numbers and letters are provided.

O

Octal — a numbering system which uses eight as the base, and the digits 0, 1, 2, 3, 4, 5, 6 and 7. The Octal system is not used very much nowadays in microcomputer fields. The Hexadecimal system is more common (see Hexadecimal).

Operating System — the software or firmware generally provided with the machine that allows you to run other programs.

OR — an arithmetic operation that returns a 1, if one or more inputs are 1.

Oracle — a method of sending text messages with a broadcast television signal. A teletext set is required to decode the messages. Oracle is run by Independent Television Service in the UK, and a similar service — Ceefax — is provided by the BBC.

Output — information or data fed out by the computer to such devices as a TV-like screen, a printer or a cassette tape. The output usually consists of the information which the computer has produced as a result of running a program.

Overflow — a number too large or too small for the computer to handle.

P

Pad — see Keypad.

Page — often used to refer to the amount of information needed to fill one TV screen, so you can talk about seeing a page of a program, the amount of the listing that will appear on the screen at one time.

PASCAL — a high level language.

Peripheral — anything which is hooked onto a computer, for control by the computer, such as a disk unit, a printer or a voice synthesiser.

Port — a socket through which information can be fed out of or in to a computer.

Prestel — the British telecom name for a system of calling up pages of information from a central computer via the computer, which tells the computer how to make decisions and how to carry out arithmetic operations. This memory is unaffected when you turn the computer off. The second type is RAM (random access memory). This memory holds the program you type in at the keyboard or send in via a cassette or disk. In most computers the computer 'forgets' what is in RAM when you turn the power off.

Microprocessor — the heart of any computer. It requires peripheral unit interfaces, such as a power supply and input and output devices, to act as a microcomputer.

MODEM — stands for Modulator Demodulator. This is a device which allows two computers to talk to each other over the telephone. The computers usually use a cradle in which a telephone receiver is placed.

Monitor — this has two meanings in computer terms. One meaning is a television-like display. A monitor has no facility for tuning television programs, and usually the picture produced on a monitor is superior to that produced by an ordinary television. The second meaning of a monitor relates to ROM. The monitor of a computer is described as the information it has built in when you buy it. This information allows it to make decisions and carry out arithmetic computations.

Motherboard — a framework to which extra circuits can be added. These extra circuits often give the computer facilities which are not built-in, such as that of producing sound or of controlling a light pen.

MPU — an abbreviation for Microprocessor Unit.

N

Nano-second — a nano-second is one thousand billionth of a second, the unit of speed in which a computer or a memory chip is often rated.

Non-Volatile Memory — memory which is not lost when the computer is turned off. Some of the smaller computers such as the TRS-80 Pocket Computer have non-volatile memory. The batteries hold the program you enter for several hundred hours.

Not — a Boolean logic operation that changes a binary digit into its opposite.

Null String — a string which contains no characters. It is shown in the program as two double quote marks, without anything between them.

telephone and displaying them on a television screen. A similar commercial version in the United States is known as The Source.

Program — in computer terms program has two meanings. One is the list of instructions that you feed into a computer, and the second is used as a verb, as in 'to program a computer'.

PROM — stands for Programmable Read Only Memory. This is a device which can be programmed, and once it is then the program is permanent (also see EPROM and ROM).

R

Random Access Memory (RAM) — the memory within a computer which can be changed at will by the person using the computer. The contents of RAM are usually lost when a computer is turned off. RAM is the memory device that stores the program that you type in and also stores the results of calculations in progress.

Read-Only Memory (ROM) — in contrast to RAM, information in ROM cannot be changed by the user of the computer, and the information is not lost when the computer is turned off. The data in ROM is put there by the manufacturers and tells the computer how to make decisions and how to carry out arithmetic computations. The size of ROM and RAM is given in the unit K (see K).

Recursion — the continuous repetition of a part of the program.

Register — a specific place in the memory where one or more computer words are stored during operations.

Reserved Word — a word that you cannot use for a variable in a program because the computer will read it as something else. An example is the word TO. Because TO has a specific computer meaning, most computers will reject it as a name for a variable. The same goes for words like FOR, GOTO and STOP.

Routine — this word can be used as a synonym for program, or can refer to a specific section within a program (also see Subroutine).

S

Second Generation — this has two meanings. The first applies to computers using transistors, as opposed to first generation computers which used valves. Second generation can also mean the second copy of a particular program; subsequent generations are degraded by more and more noise.

Semiconductor — a material that is usually an electrical insulator but under specific conditions can become a conductor.

Serial — information which is stored or sent in a sequence, one bit at a time.

Signal — an electrical pulse which is a conveyor of data.

Silicon Valley — the popular name given to an area in California where many semiconductor manufacturers are located.

SNOBOL — a high level language.

Software — the program which is entered into the computer by a user which tells the computer what to do.

Software Compatible — this refers to two different computers which can accept programs written for the other.

Static Memory — a non-volatile memory device which retains information so long as the power is turned on, but does not require additional boosts of power to keep the memory in place.

Subroutine — part of a program which is often accessed many times during the execution of the main program. A subroutine ends with an instruction to go back to the line after the one which sent it to the subroutine.

T

Teletext — information transmitted in the top section of a broadcast television picture. It requires a special set to decode it to fill the screen with text information. The BBC service is known as Ceefax, the ITV service as Oracle. Teletext messages can also be transmitted by cable, for example the Prestel service in Britain or The Source in the United States.

Teletype — a device like a typewriter which can send information and also receive and print it.

Terminal — a unit independent of the central processing unit. It generally consists of a keyboard and a cathode ray display.

Time Sharing — a process by which a number of users may have access to a large computer which switches rapidly from one user to another in sequence, so each user is under the impression that he or she is the sole user of the computer at that time.

Truth Table — a mathematical table which lists all the possible results of a Boolean logic operation, showing the results you get from various combinations of inputs.

U

UHF — Ultra High Frequency (300-3000 megaHertz).

Ultra Violet Erasing — Ultra violet light must be used to erase EPROMs (see EPROM).

V

Variable — a letter or combination of letters and symbols which the computer can assign to a value or a word during the run of a program.

VDU — an abbreviation for Visual Display Unit.

Volatile — refers to memory which 'forgets' its contents when the power is turned off.

W

Word — a group of characters, or a series of binary digits, which represent a unit of information and occupy a single storage location. The computer processes a word as a single instruction.

Word-Processor — a highly intelligent typewriter which allows the typist to manipulate text, to move it around, to justify margins and to shift whole paragraphs if necessary on a screen before outputting the information onto a printer. Word-processors usually have memories, so that standard letters and the text of letters, written earlier, can be stored.

BIBLIOGRAPHY

Compiled by Tim Hartnell

Usborne have released a number of very attractive books in their Usborne Computer Books series. Drawing on their vast experience in the field of producing low-priced, highly-coloured, attractive books for young readers, they've produced some books which will enlighten both young and not-so-young readers.

I'll look at three of their titles, three which cover just about the whole field of computer interests:

Information Revolution
(Lynn Myring and Ian Graham, Rigby).
Presenting an eminently readable introduction to the 'revolution' which covers such fields as computers (of course), text information services via the television screen, word processing, 'future phones' and satellite communications, *Information Revolution* is an ideal guide for the person who wants an easy-to-read introduction to the field.

Computer Jargon
(Corinne Stockley and Lisa Watts).
The tone of this book is set by the frontispiece, which has a number of odd little coloured robots sitting around a table laden with computer junk, pointing at each piece saying "This is a disk drive", "This is a digital tracer" (!) and "This is a printer".

Robotics — What Robots Can Do and How They Work
(Tony Potter and Ivor Guild).
This is definitely a candidate for the award of 'the longest title of the year'. But it is very accurate. Don't be put off by the pretty pictures, as you'll soon discover this book has a lot of solid information. Topics covered include "What robots can and cannot do", "How arm robots work", "How to teach a robot" and "Build your own micro-robot"; this last section actually includes nine pages of circuit diagrams and all to build a little two-motor robot which, following a program typed into your micro, will run about the floor. Robotics is a field of the near future (with personal robots certain to be a bigger craze — when 'real robots' finally arrive — than computers will ever be).

Practise Your BASIC
(Gaby Waters and Nick Cutler).
You'll find this book — which predictably contains a number of exercises, puzzles and problems to solve by writing programs — should be useful in giving you a number of 'core problems' which will run on your computer and which can then be modified to take advantage of your system's special features. Program listings include 'Pattern Puzzles', 'Jumping Man', 'Horse Race', 'Word Editor' and 'Treasure Hunt', a mini-Adventure.

Help With Computer Literacy
(June St Clair Atkinson, Houghton Mifflin).
This is a large format book with an attractive cover, fairly priced for its 122 pages. It appears to be aimed at the early to middle years of secondary education, but contains a lot of material which those teaching younger children could easily adapt. Although it avoids the 'Gee Whiz' approach of the Usborne texts, it uses cartoons and diagrams to get its message across in an inviting manner.

The Interface Computer Encyclopedia
(Ken Ozanne, Interface Publications).
Compiled by a lecturer in mathematics at the NSW Institute of Technology, this work could perhaps be more accurately called 'The Computer Book of Lists', rather than an encyclopedia. It contains annotated references to 'all' microprocessors, 'all' microcomputers, and 'most' microcomputing magazines. The inverted commas are there because — as the author admits candidly in his introduction — any such work is likely to be out of date even before it is published. Fat (445 pages) with minimalist presentation (the whole book is dumped directly from a word processor onto a dot-matrix printer) you'll find this a useful work if you want a ready reference to chips, computers and the ever-growing field of specialist magazines.

Computer Resource Book — Algebra
(Thomas Dwyer and Margot Critchfield, Houghton Mifflin).
Dwyer and Critchfield have clocked up an enviable string of successful computer books, and this one, part of a series, shows why. With simple, but valuable programs, the authors lead the reader (who can be a secondary student, or an instructor) through most of the phrases of the BASIC programming language which are common to all low-priced computers, and most educational time-sharing systems.

Apple II BASIC
(David Goodfellow, Tab Books Inc.).
Attractively packaged, this book is clearly laid out, with an abundance of example programs; it takes a commendable approach to the business of teaching programming, with the qualities of 'programming style' introduced without fanfare. In the crowded field of 'how to program your Apple' books, this one stands out. Much of the material presented is applicable to any microcomputer.

Pre-Computer Activities
(Dorothy Diamond, Hulton Educational).
This practical guide for teachers and parents can help make children familiar with essential computer processes and language before they have hands-on experience. The book contains a number of interesting activities, including investigating binary numbers using little lights, and working with cardboard 'calculators' before getting to the real thing. The discussion on computer graphics is enlivened by reference to the solid blocks which make up a 'Pacman' figure.

Word Processing Experience
(Janet Pigott and Roger Atkins-Green, Stanley Thornes Publishers Ltd.).
Designed for schools, but ideal for adapting if you'd like to increase your skill with a word processor (or simply because you'd like to see what word processors can do so you can write one for your own microcomputer), this book looks at the mechanics of word-processing, while passing on a great deal of useful information about word-processing techniques.

An Introduction to Micro-electronics and Microprocessor Systems
(G H Curtis and P G Wilks, Stanley Thornes, Publishers Ltd.).
This work was written for junior college students and older school pupils, as well as for non-specialists who wanted a comprehensive — if dry — technical introduction to the subject. The going is not easy, but it's worth the effort. Topics covered include 'Logic', 'Programming the Microcomputer' and 'Analogue, Binary and Digital Systems'.

Computer Images — State of the Art
(Joseph Deken, Thames and Hudson).
This is a beautiful book, large and glossy, and packed with quality full-colour computer-generated (or, in some cases, computer-modified) images. The whole fascinating field of modern computer graphics is discussed — from television pro-

gramme introductions using photographs which are colour-modified, twisted and tweeked, to the use of incredible high-resolution images in simulators for flight training and tank manoeuvring. You'll read (and see) how computers are used to produce images, how these are used for education and communication, why 'art for art's sake' is a goal worth pursuing, and how computer images can evolve using processes uncannily akin to the processes by which groups of cells multiply and divide. If you want to see what can be done with high resolution graphics and when time, money and skill abound, you should get this book.

Computer Bluff
(Stephen Castell, Quartermaine House Ltd.).
A much more valuable book than its title indicates, it contains a lot of information on the what and how of computers, along with a generous dollop of computer jargon (or 'How to Cheat in Computer-Speak'). The style is gentle and amusing, with no appalling puns or excessive asides (such as 'didja get that joke, buster?'). A pleasant, painless book which you can digest, then give to a parent.

Penguin Books has moved into the computer field with enthusiasm. As well as a 'Getting the Most Out of Your...' series, they have a number of games books. Two which stand out are **The Penguin Book of VIC 20 Games** (Paul Copeland) and **The Penguin Book of Commodore 64 Games** (Robert Young and Paul Copeland). Priced at £4.95 each, these large format books include such programs as 'Space Venture', 'Oil Rig' and 'Red Alert'. Worth buying, even if you do not have a VIC or a Commodore 64, simply as a source of ideas for new programs to create on your own microcomputer.

Arcade Games for Your VIC 20 and **Arcade Games for Your Commodore 64** (Brett Hale, Corgi/Addison-Wesley) by contrast, are definitely only for those who have the machine specified. The programs are locked irrevocably to the computer named. Taking advantage of a number of machine-specific features (such as sprite graphics on the 64), Brett has produced a selection of around 20 programs for each machine. Each one is listed twice, the first time for the joystick and the second time for the keyboard. Titles include 'Galaxy Robbers', 'Bullet Heads' and 'Yackman'.

CREATING ADVENTURE PROGRAMS

There are a number of books, some of which are aimed at computer owners, which will help you if you are one of the many, many computer games players who are interested in developing 'Adventure' and 'Dungeons' type programs. The place to start is with TRS Hobbies' **Dungeons and Dragons** (TM) Basic Set, which comes with the introductory rule book, Dungeon Dice (tm) and an instruction module, along with a sample scenario 'The Keep on the Borderlands'. If you're new to the field, you should start with this set to give you an idea how 'real life' Adventure programs are built up.

Additional information is provided by **Fantasy Role-Playing Games** (J. Eric Holmes, Hippocrene Books Inc.) which looks at the whole field and, despite some disparaging things to say on computer versions of such games, is worth looking for. Another overview of the field — with more sympathetic comments on the use of computers — is provided by **Dicing With Dragons — An Introduction to Role-Playing Games** (Ian Livingstone, Routledge and Kegan Paul), which includes a full 'solo Adventure', a review of the major games on the market, and a fascinating chapter on the pleasures and perils of being Dungeon Master in 'Playing God'.

Fantasy Wargaming (compiled Bruce Galloway, published Patrick Stephens) provides a complete unified system for 'historically accurate' (or at least in tune with the beliefs and circumstances of individuals in the peasant, feudal-economy times in which many Adventures are set) games. The fight, weapon and monster tables alone are worth the book, as many of their ideas can easily be incorporated into your Adventures.

There are two computer Adventure books which you could get to help you in the fascinating area of producing Adventure games on your machine.

Creating Adventure Programs on Your Computer
(Andrew Nelson, Interface Publications).
Written by the author of *More Games for Your VIC20* and *Games for Your TI 99/4A*, in the Virgin Books games series, this book takes you through the task of developing an Adventure program of your own, concentrating more on the 'Loot and Pillage' school of gaming than the Scott Adams' 'solve this puzzle to advance' field. Three complete Adventure programs are included.

Write Your Own Adventure Programs for Your Microcomputer (Jenny Tyler and Les Howarth, Usborne) is a much quicker introduction to the field than Nelson's, but nevertheless packs a lot of valuable information into its 48 pages. Step-by-step instructions are provided for creating an Adventure from scratch. A complete program — 'Haunted House' — is included in the book.

The Age of Computers is the general title of four fine books produced by Wayland Publisher Limited. Each priced at £4.95, the books present a careful, but inviting, view of four aspects of the computer field, one on the history of computers and the others looking at specific areas of modern computer application. Each book is by Ian Litterick and Chris Smithers. The four titles are **The Story of Computers,** with Charles Babbage and Uncle Sir Clive Sinclair just inside the cover (and these two pictures accurately sum up the historical period covered by the book); **How Computers Work** (with chapter headings including 'Bits, Bytes and Binary', 'Decision-making by Transistor', and 'Talking With Computers'); **Computers in Everyday Life** (such things as 'Robots in the Home', 'Magnetic Money' and 'Medicine and the Disabled'); and **Computers and You** ('Computopia', 'Big Brother', 'War and Peace' and — a fascinating final chapter — 'Will Computers Need Us?').

Inside BASIC Games
(Richard Mateosian, Sybex).
This book is a slightly overwritten guide to understanding computer games. You'll learn how to write interactive programs in BASIC and how the principles of system development are applied to small computers. The book also looks at how the features of specific small computer systems have been supported in BASIC. If you can contend with the verbiage, you'll find this book well worthwhile.

1001 Things to Do With Your Personal Computer
(Mark Sawush, Tab Books).
Big and fat, and full of ideas, you'll find much here of interest to enlarge your computer horizons. The book tells you about writing music and stories with your computer, aiding a mechanic or a carpenter, solving simultaneous equations, astrology and much, much more.

Stimulating Simulations
(C.W. Engel, Hayden Book Company).
Here are 12 unique programs written in a good, general version of BASIC. The fascinating programs include 'Forest Fire', 'Rare Birds' and 'The Devil's Dungeon'. You're sure to enjoy playing those three, along with 'Diamond Thief', in which the computer decides who has committed the crime, then challenges you to discover which of the suspects is guilty. The material in this book is generally tightly programmed, and can be a helpful source of ideas to improve your own computer work.

The BASIC Handbook
(David A. Lien, Compusoft Publishing).
This is an encyclopedia of the BASIC language. It comes into its own when you find a program in a magazine or book which you'd love to try, but are frustrated because it is written for another version of BASIC. Every BASIC word you've ever heard of (and many you may not have, such as NE, GOTO-OF and LE) is in here, along with a number of variations, one of which will almost certainly be on your machine.

BASIC Computer Games
(David Ahl, Creative Computing Press).
This is a classic work, still selling well despite the fact it was one of the first such books — if not *the* first — on the market. David Ahl has been in personal computers even before there were such things. Although several of the games are overly-dependent on the random number generator, you'll find there are many, many games you'll want to adapt and improve for your own computer.

How to Buy (and Survive) Your First Computer
(Carolee Nance Kolve, McGraw-Hill Book Company).
When is a business ready for a computer? How do you make an intelligent, informed choice among the hundreds of computers available? Will a computer improve a company's operations? Answers to these and a score of similar questions are in this book, which explains in detail what to consider before buying, how to select the right computer, and what to do after ordering the computer to ensure a successful installation. Ms Kolve has over 15 years computer experience (including a stint with IBM) and brings her experience to bear in a relatively easily-digestible guide.

Your First BASIC Program
(Rodnay Zaks, Sybex).
This book, liberally illustrated with large red dinosaurs in a variety of situations vaguely related to the text (one, for instance, as a cowboy getting tangled up in his ropes with the caption 'Be careful when looping'), is a gentle and worthwhile introduction to the not-so-secret secrets of programming in BASIC. When you want to move beyond just typing in other people's programs from books and magazines, this may be a good place to start.

This bibliography was compiled by the series editor, Tim Hartnell, who has felt constrained not to recommend any of his own books. However, he asked us to mention two which could be of use and interest to you.

The first is **The Personal Computer Guide** (Virgin Books) which explains what a personal computer is, and answers questions like "Will it help my kids?", "What sort of games can we play on it?" and "What can I use it for in the home?". The book describes many of the most popular computers available today, with illustrations, technical specifications and other information to help you to choose the equipment best suited to your requirements. Also included is an introduction to BASIC programming, with details of programs suitable for use in the home, a list of suppliers and user clubs, and a guide to further reading. There are also chapters covering the personal computer's history and its future. When you're ready to upgrade, you'll find this book a good, unbiased, reference work which looks at the choices facing you.

Tim Hartnell's Giant Book of Computer Games.
Described by *Personal Computer News* as 'a good source of ideas', this 386-page book, published by Fontana, for £3.95, contains over 40 programs which will run with minimum modifications on most popular microcomputers. The games include chess (of a sort!), a 17K Adventure and 'Hyperwar'.

The Virgin Computer Games Series

GAMES FOR YOUR ATARI 600 XL

More than 20 challenging programs, each one especially written for the series and guaranteed to provide hours of entertainment.

SPACE ATTACK (defend your space station, while warding off pesky alien invaders); CHOPPER MISSION (you are at the controls of a rescue helicopter – the fate of imprisoned hostages depends on you); BREAKTHROUGH (test your co-ordination and nerve in this fast-moving version of 'Breakout'); ASTEROID STORM (take control of your trusty spacecraft as you weave in and out of a rock storm); and the chance to escape through a crowded minefield to safety beyond the yellow brick wall.

GAMES FOR YOUR ATARI will improve your programming skills as you follow the instructions to put each of the programs into your machine, and comes complete with a brief dictionary of computer terms, a selective bibliography and some hints on how to extend the programs in the book.

Programs of originality and quality for all the family.

ISBN 0 86369 037 8
United Kingdom £2.95
Australia $9.95 (recommended)

THE VIRGIN COMPUTER
GAMES SERIES

GAMES FOR YOUR ATARI 600XL